# 1. DESCRIPTION

The M306H7MG/MC-XXXFP and M306H7FGFP are single-chip microcomputers using the high-performance silicon gate CMOS process using M16C/62 Series CPU core and is packaged in a 100-pin plastic molded QFP. This single-chip microcomputer operates using sophisticated instructions featuring a high level of instruction efficiency. With 1M bytes of address space, this is capable of executing instructions at high speed. This also features a built-in data slicer, making this correspondence to Global broadcasting service.

## 1.1 Features

- Memory capacity ..... ROM Mask version : 256 K/128 K bytes
  Flash memory version : 256 K bytes
  RAM Mask version : 8 K/5 K bytes
  Flash memory version : 8 K bytes
- Shortest instruction execution time .. 62.5 ns ($f(X_{IN})$=16 MHz)
- Supply voltage ..... $V_{CC1}$=3.00 V to $V_{CC2}$, $V_{CC2}$=4.5 V to 5.5 V (at $f(X_{IN})$=16 MHz)
  $V_{CC1}$=2.00 V to $V_{CC2}$, $V_{CC2}$=2.00 V to 5.5 V (at $f(X_{CIN})$=32 kHz)
  *$V_{CC2}$=2.0 V to 2.9 V: Operates only in the low power dissipation mode
- Interrupts ..... 25 internal and 8 external interrupt sources, 4 software interrupt sources; 7 levels
- Multifunction 16-bit timer ..... 5 output timers + 6 input timers
- Serial I/O ..... 6 channels
  UART/clock synchronous: 3
  Clock synchronous: 2
  Multi-master $I^2C$: 1
- DMAC ..... 2 channels (trigger: 24 sources)
- A/D converter ..... 8 bits X 8 channels (Expandable up to 10 channels)
- CRC calculation circuit ..... 1 circuit
- Watchdog timer ..... 1 line
- Programmable I/O ..... 79 lines (P6 to P7, $P8_0$ to $P8_4$: Can be used as 3.3 V interface)
- Input port ..... 1 port ($P8_5$ shared with $\overline{NMI}$ pin)
- Clock generating circuit ..... 2 built-in circuits
  (built-in feedback resistor, external crystal oscillator is required)
- Data slicer ..... For PDC, VPS, WSS, EPG-J, CC, CC2X, ID-1

## 1.2 Applications

DVD recorder, HDD recorder

------Table of Contents------


**1. DESCRIPTION ..... 1**
1.1 Features ..... 1
1.2 Applications ..... 1
Table of Contents ..... 2
1.3 Pin Configuration ..... 3
1.4 Performance Outline ..... 4
1.5 Block Diagram ..... 6
1.6 Memory ..... 10
**2. CENTRAL PROCESSING UNIT (CPU) ..... 11**
2.1 Data Registers (R0, R1, R2 and R3) ..... 11
2.2 Address Registers (A0 and A1) ..... 11
2.3 Frame Base Register (FB) ..... 12
2.4 Interrupt Table Register (INTB) ..... 12
2.5 Program Counter (PC) ..... 12
2.6 User Stack Pointer (USP) and Interrupt Stack Pointer (ISP) ..12
2.7 Static Base Register (SB) ..... 12
2.8 Flag Register (FLG) ..... 12
**3. RESET ..... 13**
3.1 Hardware Reset ..... 13
3.2 Software Reset ..... 14
3.3 Watchdog Timer Reset ..... 14
3.4 SFR ..... 17
**4. CLOCK GENERATION CIRCUIT ..... 25**
4.1 Oscillator Circuit ..... 30
4.2 CPU Clock and Peripheral Function Clock ... 33
4.3 Clock Output Function ..... 33
4.4 Power Control ..... 35
4.5 System Clock Protective Function ..... 41
**5. PROTECTION ..... 42**
**6. INTERRUPTS ..... 43**
6.1 Type of Interrupts ..... 43
6.2 Software Interrupts ..... 44
6.3 Hardware Interrupts ..... 45
6.4 Interrupts and Interrupt Vector ..... 46
6.5 Interrupt Control ..... 48
6.6 I Flag ..... 50
6.7 IR Bit ..... 50
6.8 ILVL2 to ILVL0 Bits and IPL ..... 50
6.9 Interrupt Sequence ..... 51
6.10 Interrupt Response Time ..... 52
6.11 Variation of IPL when Interrupt Request is Accepted ..... 52
6.12 Saving Registers ..... 53
6.13 Returning from an Interrupt Routine ..... 55
6.14 Interrupt Priority ..... 55
6.15 Interrupt Priority Resolution Circuit ..... 55
6.16 INT Interrupt ..... 57
6.17 NMI Interrupt ..... 58
6.18 Address Match Interrupt ..... 58
**7. WATCHDOG TIMER ..... 60**
**8. DMAC ..... 62**
8.1 Transfer Cycles ..... 67
8.2 Number of DMA Transfer Cycles ..... 69
8.3 DMA Enable ..... 70
8.4 DMA Request ..... 70
8.5 Channel Priority and DMA Transfer Timing ... 71
**9. TIMERS ..... 72**
9.1 Timer A ..... 73
9.2 Timer B ..... 87
**10. SERIAL I/O ..... 93**
10.1 UARTi (i=0 to 2) ..... 93
10.2 Clock Synchronous serial I/O Mode ..... 102
10.3 Clock Asynchronous Serial I/O (UART) Mode .... 109
10.4 Special Mode 1 (I2C mode) ..... 116
10.5 Special Mode 2 ..... 126
10.6 Special Mode 3 (IE mode) ..... 131
10.7 Special Mode 4 (SIM Mode) (UART2) ..... 133
10.8 SI/O3 and SI/O4 ..... 138
**11. MULTI-MASTER I2C BUS INTERFACE ..... 143**
**12. A/D CONVERTER ..... 163**
12.1 One-shot Mode ..... 167
12.2 Repeat mode ..... 169
12.3 Single Sweep Mode ..... 171
12.4 Repeat Sweep Mode 0 ..... 173
12.5 Repeat Sweep Mode 1 ..... 175
12.6 Sample and Hold ..... 177
12.7 Extended Analog Input Pins ..... 177
12.8 External Operation Amp Connection Mode .... 177
12.9 Current Consumption Reducing Function ..... 178
12.10 Analog Input Pin and External Sensor Equivalent Circuit Example .... 178
12.11 Caution of Using A/D Converter ..... 179
**13. CRC CALCULATION ..... 180**
**14. EXPANSION FUNCTION ..... 182**
14.1 Expansion function description ..... 182
14.2 Expansion memory ..... 183
14.3 slice RAM ..... 184
14.4 CRC Operation Circuit (EPG-J) ..... 187
14.5 Expansion Register ..... 200
14.6 Expansion Register Construction Composition ... 240
14.7 8/4 Humming Decoder ..... 247
14.8 24/18 Humming Decoder ..... 248
14.9 I/O Composition of pins for Expansion Function . 250
**15. PROGRAMMABLE I/O PORTS ..... 252**
15.1 Port Pi Direction Register (PDi Register, i = 0 to 9) ..252
15.2 Port Pi Register (Pi Register, i = 0 to 9) ..... 252
15.3 Pull-up Control Register 0 to Pull-up Control Register 2 (PUR0 to PUR2 Registers) ..... 252
15.4 Port Control Register ..... 252
**16. ELECTRICAL CHARACTERISTICS ..... 263**
**17. FLASH MEMORY VERSION ..... 279**
17.1 Flash Memory Performance ..... 279
17.2 Memory Map ..... 281
17.3 Boot Mode ..... 282
17.4 Functions To Prevent Flash Memory from Rewriting ... 282
17.5 CPU Rewrite Mode ..... 284
17.6 Data Protect Function ..... 298
17.7 Status Register ..... 298
17.8 Full Status Check ..... 300
17.9 Standard Serial I/O Mode ..... 302
17.10 Parallel I/O Mode ..... 307
**18. PACKAGE OUTLINE ..... 308**
**19. USEGE NOTES ..... 309**
19.1 Precautions for Power Control ..... 309
19.2 Precautions for Protect ..... 309
19.3 Precautions for Interrupts ..... 309
19.4 Precautions for DMAC ..... 313
19.5 Precautions for Timers ..... 314
19.6 Precautions for Serial I/O (Clock-synchronous Serial I/O) ... 317
19.7 Precautions for Serial I/O (UART Mode) .... 318
19.8 Precautions for A/D Converter ..... 318
19.9 Precautions for Programmable I/O Ports .... 318
19.10 Electric Characteristic Differences Between Mask ROM and Flash Memory Version Microcomputers ..... 318
19.11 Precautions for Flash Memory Version ..... 318
19.12 Other Notes ..... 323
19.13 Serial I/O (RxDi input setup time) ..... 325
19.14 Precautions for LP3 and LP4 pins ..... 326

## 1.3 Pin Configuration

Figures 1.1 shows the pin configuration (top view).

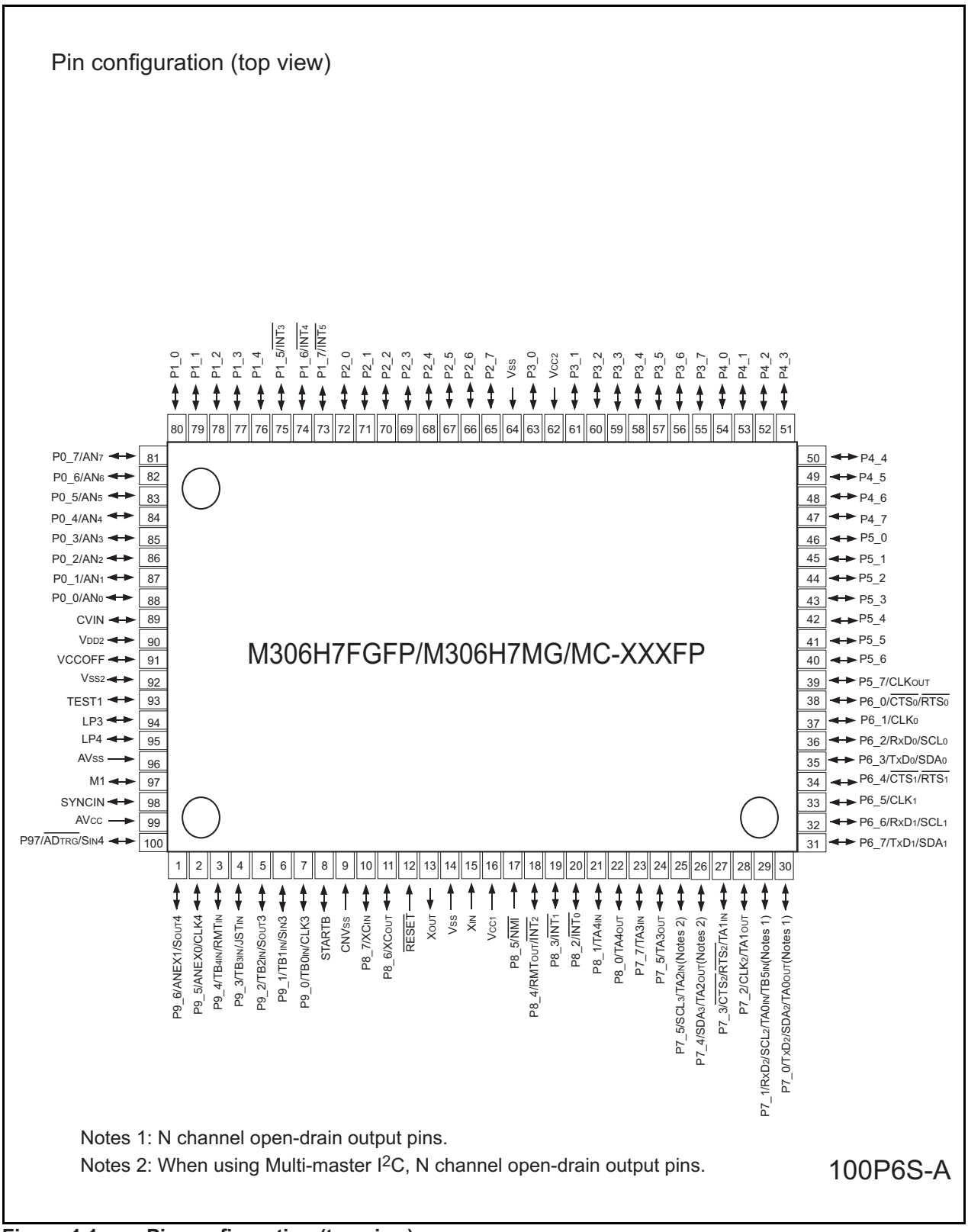



**Figure 1.1 Pin configuration (top view)**

## 1.4 Performance Outline

Performance outline is shown in Table 1.1.

**Table 1.1 Performance outline**

| Item | | Performance |
|---|---|---|
| Number of basic instructions | | 91 instructions |
| Shortest instruction execution time | | 62.5 ns (f(XIN)= 16MHz, VCC= 4.5V to 5.5V) |
| Memory capacity | ROM | Refer to the Product table (Table 1.2) |
| | RAM | Refer to the Product table (Table 1.2) |
| I/O port | P0 to P5, P86 to P87, P9 | 8-bit x 7, 2-bit x 1 : VCC2 system |
| | P6 to P7, P80 to P84 | 8-bit x 2, 5-bit x 1 : VCC1 system |
| Input port | P85 | 1-bit x 1 (NMI pin VCC2 level judgment) : VCC2 system |
| Multi function timer | TA0, TA1, TA2, TA3, TA4 | 16-bit x 5 channels |
| | TB0, TB1, TB2, TB3, TB4, TB5 | 16-bit x 6 channels |
| Serial I/O | | 3 channels<br>Clock synchronous serial I/O,<br>Clock asynchronous serial I/O,<br>I2C bus[1], or IEBus[2]<br><br>2 channels<br>Clock synchronous serial I/O |
| Serial I/O | Multi-master I2C | I2C bus x 1 |
| A/D converter | | 8 bits x (8 + 2) channels |
| DMAC | | 2 channels (trigger: 24 sources) |
| CRC calculation circuit | | CRC-CCITT |
| Watchdog timer | | 15 bits x 1 (with prescaler) |
| Interrupt | | 25 internal and 8 external sources, 4 software sources, 7 levels |
| Clock generation circuit | | 2 circuits<br>• Main clock } (These circuits contain a built-in feedback resistor and external crystal oscillator)<br>• Sub-clock } |
| Power supply voltage | | VCC1=3.00 V to VCC2, VCC2= 4.5 V to 5.5 V (at f(XIN)=16MHz) |
| | | VCC1=3.00 V to VCC2, VCC2= 4.00 V to 5.5 V (at f(XIN)=16MHz)[3] |
| | | VCC1=2.90 V to VCC2, VCC2= 2.90 V to 5.5 V (at f(XIN)=16MHz, at divide-by-8 or 16)[3] |
| | | VCC1=2.0 V to VCC2, VCC2=2.0 V to 5.5 V (at f(XCIN)=32kHz, only low-power consumption mode) [3,4] |
| Flash memory | Program/erase voltage | 5.0 V ± 0.25 V |
| | Number of program/erase | 100 times |
| Device configuration | | CMOS high performance silicon gate |
| Package | | 100-pin plastic mold QFP |
| Data slicer | Slice RAM | 864 bytes (48 x 18 x 8-bit) |
| | Data slicer | Corresponds to PDC, VPS, WSS, EPG-J, CC, CC2X and ID-1 |

NOTES:
1. I²C bus is a registered trademark of Koninklijke Philips Electronics N.V. If you desire this option, please so specify.
2. IEBus is a registered trademark of NEC Electronics Corporation.
3. If the $V_{CC2}$ supply voltage is less than 4.50 V, the A/D converter, data slicer cannot be used.
4. If the $V_{CC2}$ supply voltage is less than 2.60 V, be aware that only the CPU, RAM, clock timer, interrupt, and I/O ports can be used. Other control circuits (e.g., timers A and B, serial I/O, UART) cannot be used.

**Table 1.2 Product table**

| Type No. | ROM capacity | RAM capacity | Package type | Remarks |
|---|---|---|---|---|
| M306H7MG-XXXFP | 256K bytes | 8K bytes | 100P6S-A | Mask ROM version |
| M306H7MC-XXXFP | 128K bytes | 5K bytes | | Mask ROM version |
| M306H7FGFP | 256K bytes | 8K bytes | | Flash Memory version |

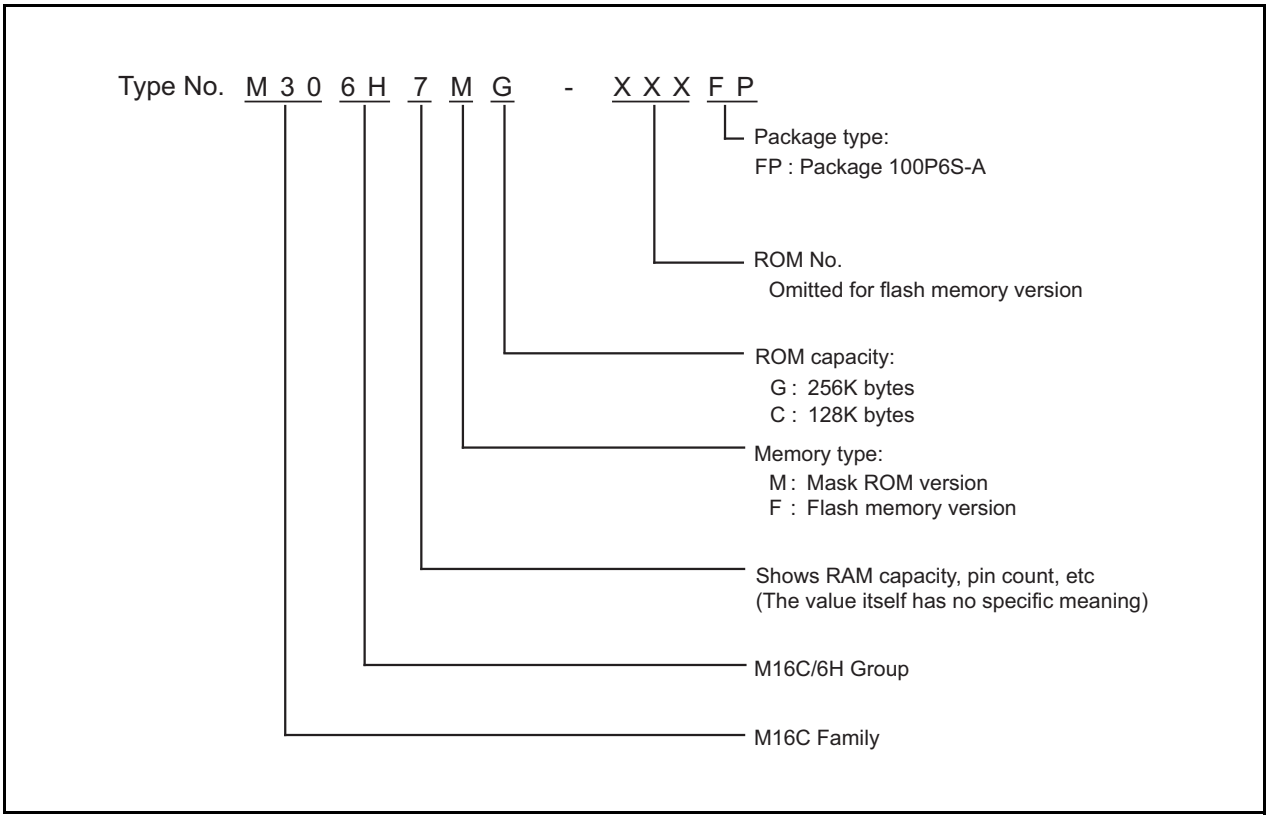



**Figure 1.2 Type No, Memory Size, and Package**

## 1.5 Block Diagram

Figure 1.3 is a block diagram.

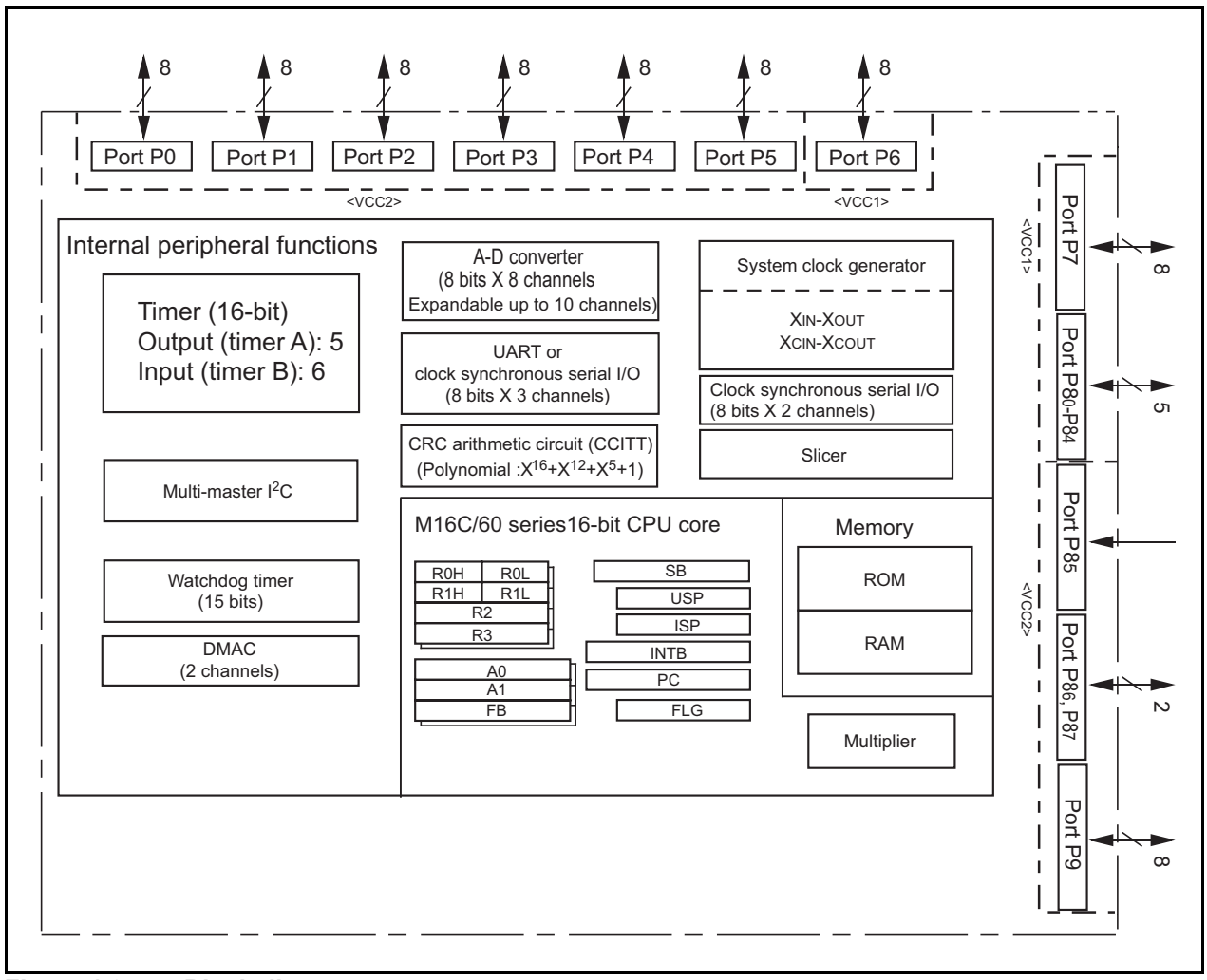



**Figure 1.3 Block diagram**

**Table 1.3 Pin Description (1)**

| Pin name | Signal name | I/O type | Power supply | Function |
|---|---|---|---|---|
| $V_{CC1}$, $V_{CC2}$, $V_{SS}$ | Power supply input | | | Apply 2.00 V to 5.5 V to the Vcc1 and Vcc2 pins. Apply 0 V to the Vss pin. Input condition of $V_{CC1}$ and $V_{CC2}$ are $V_{CC1} \leq V_{CC2}$. (1.) |
| $CNV_{SS}$ | $CNV_{SS}$ | Input | $V_{CC2}$ | Connect this pin to $V_{SS}$. |
| $\overline{RESET}$ | Reset input | Input | $V_{CC2}$ | "L" on this input resets the microcomputer. |
| $X_{IN}$<br>$X_{OUT}$ | Clock input<br>Clock output | Input<br>Output | $V_{CC2}$ | These are I/O pins provided for main clock oscillation circuit.<br>Connect ceramic resonator or crystal oscillator between pins $X_{IN}$ and $X_{OUT}$. To use an externally derived clock, input it to $X_{IN}$ pin and leave $X_{OUT}$ pin open. |
| $AV_{CC}$ | Analog power supply input | | | This pin is a power supply input for the A/D converter. Connect this pin to $V_{CC}$. |
| $AV_{SS}$ | Analog power supply input | | | This pin is a power supply input for the A/D converter. Connect this pin to $V_{SS}$. |
| $P0_0$ to $P0_7$ | I/O port P0 | Input/output | $V_{CC2}$ | This is an 8-bit CMOS I/O port. This port has an I/O select direction register, allowing each pin in that port to be directed for input or output individually.<br>If any port is set for input, selection can be made for it in a program whether or not to have a pull-up resistor in 4 bit units. Pins in this oprt also function as A/D converter input pins as selected by Program. |
| $P1_0$ to $P1_7$ | I/O port P1 | Input/output | $V_{CC2}$ | This is an 8-bit I/O port equivalent to P0. Pins $P1_5$ to $P1_7$ in this port also functionas $\overline{INT}$ interrupt input pins as selected by software. |
| $P2_0$ to $P2_7$ | I/O port P2 | Input/output | $V_{CC2}$ | This is an 8-bit I/O port equivalent to P0. |

Note 1: In this datasheet, hereafter, $V_{CC}$ refers to $V_{CC2}$ unless otherwise noted.

**Table 1.4 Pin Description (2)**

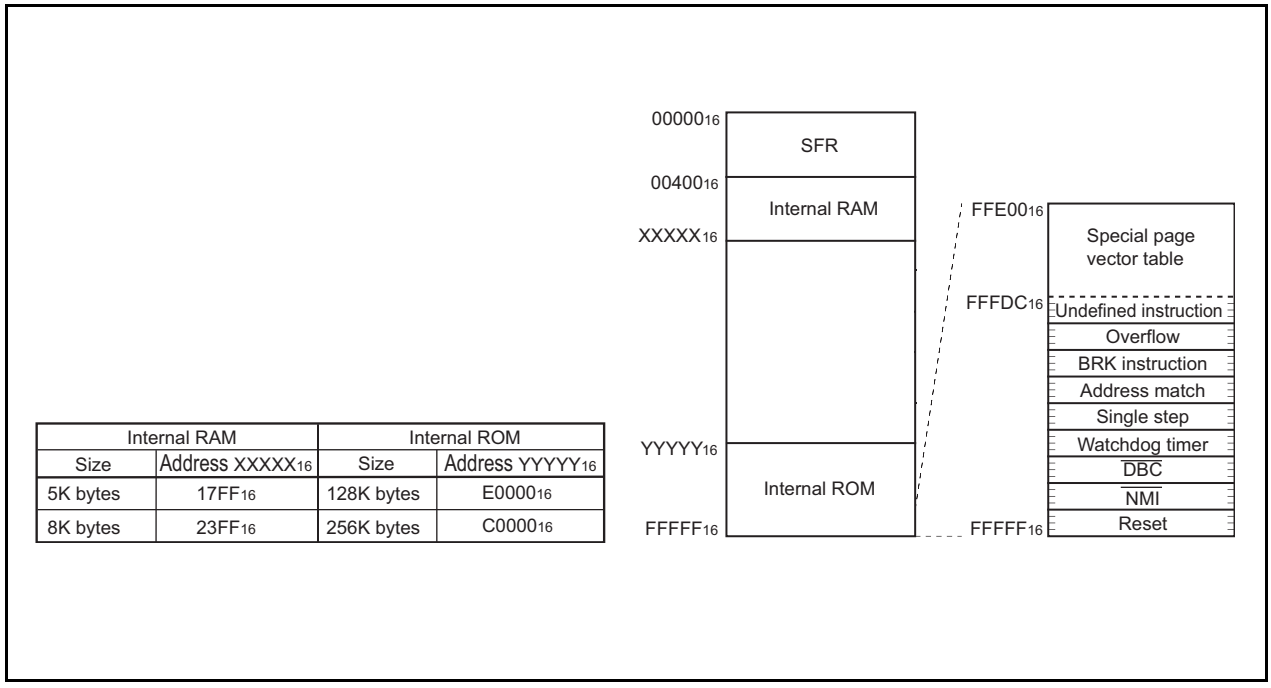
| Pin name | Signal name | I/O type | Power supply | Function |
|---|---|---|---|---|
| $P3_0$ to $P3_7$ | I/O port P3 | Input/output | $V_{CC2}$ | This is an 8-bit I/O port equivalent to P0. |
| $P4_0$ to $P4_7$ | I/O port P4 | output | $V_{CC2}$ | This is an 8-bit I/O port equivalent to P0. |
| $P5_0$ to $P5_7$ | I/O port P5 | Input/output | $V_{CC2}$ | This is an 8-bit I/O port equivalent to P0.<br>The same frequency as divide-by-8, 32 of $X_{IN}$ from P57 or $X_{CIN}$ are output by program selecting. |
| $P6_0$ to $P6_7$ | I/O port P6 | Input/output | $V_{CC1}$ | This is an 8-bit I/O port equivalent to P0.<br>These pins function as I/O pin of UART0 and UART1 by selecting it by the program. |
| $P7_0$ to $P7_7$ | I/O port P7 | Input/output | $V_{CC1}$ | This is an 8-bit I/O port equivalent to P0 ($P7_0$ and $P7_1$ are N channel open-drain output). This port can function as I/O pins for timers $A_0$ to $A_3$ when so selected in a program.<br>Furthermore, $P7_0$ to $P7_3$ function as I/O pins of UART2, $P7_1$ function as input pin of timer B5, and $P7_4$ and $P7_5$ function as I/O pin of multi-master $I^2C$ bus. |
| $P8_0$ to $P8_4$ | I/O port<br>$P8_0$ to $P8_4$ | Input/output | $V_{CC1}$<br>($P8_0$ to $P8_4$) | $P8_0$ to $P8_4$, $P8_6$, and $P8_7$ are I/O ports with the same functions as P0. When selected by a program,<br>$P8_0$ to $P8_1$ function as I/O pins of timer A4, $P8_2$ to $P8_4$ function input pin of $\overline{INT}$ interrupt.<br>And, $P8_4$ also function as output pin for remote control. |

**Table 1.5 Pin Description (3)**

| Pin name | Signal name | I/O type | Power supply | Function |
|---|---|---|---|---|
| P86, P87, P85 | I/O port P86<br>I/O port P87<br>I/O port P85 | Input/output<br>Input/output<br>Input | VCC2 (P85 to P87) | P86 and P87 that when selected in a program, both can function as I/O pins for sub clock oscillation circuit. In that case, connect crystal resonator between P86 (XCOUT pin) and P87 (XCIN pin).<br>P85 is an input-only port shared with $\overline{\text{NMI}}$. $\overline{\text{NMI}}$ interrupt is generated when input on this pin changes state from high to low.<br>$\overline{\text{NMI}}$ function cannot be disabled in a program. Pull-up resistor cannot be set for this pin. |
| P90 to P97 | I/O port P9 | Input/output | VCC2 | This is an 8-bit I/O port equivalent to P0. Pins in this port also function as SI/O3 and SI/O4 of I/O pins, Timer B0 to B4 input pins, A/D converter input pins, A/D trigger input pins, or remote control input pins as selected by program. |
| VDD2, VSS2 | Power supply input | | | Analog power supply pin. Apply the same potential as VCC2 to the VDD2 pin. Apply 0 V to the VSS2 pin. |
| CVIN | Composite video signal input 1 | Input | VCC2 | This pin inputs the external composite video signal. Data-acquisition slices this signal internally by setting. |
| SYNCIN | Composite video signal input 2 | Input | VCC2 | This pin inputs the external composite video signal. Sync.-separate circuit devides this signal internally. |
| STARTB | Oscillation selection input | Input | VCC2 | This pin selects the oscillation circuit. XIN-XOUT circuit is selected when this pin is "L"; XCIN-XCOUT circuit is selected when this pin is "H". |
| LP3 | Filter output 2 | output | VDD2 | This is a filter output pin 2 (for VPS). |
| LP4 | Filter output 3 | output | VDD2 | This is a filter output pin 3 (for PDC). |
| VCC OFF | VCC1 Power supply input select | Input | VCC2 | Normally, please input "L" level. When VCC1 power supply is off, please input "H" level. |
| M1 | Mode selection input (M1 input) | Input | VCC2 | Connect it to the VSS.In the mask ROM version, connect this pin to the VSS or the VCC2. |
| TEST1 | Test input | Input | VCC2 | This is a test pin. Connect a capacitor. |

## 1.6 Memory

Figure 1.4 is a memory map of M306H7MG-XXXFP/MC-XXXFP/FCFP. The address space extends the 1M bytes from address $00000_{16}$ to $FFFFF_{16}$.
The internal ROM is allocated in a lower address direction beginning with address $FFFFF_{16}$. An internal ROM of M306H7MC-XXXFP, for instance, is allocated to the addresses from $E0000_{16}$ to $FFFFF_{16}$.
The fixed interrupt vector table is allocated to the addresses from $FFFDC_{16}$ to $FFFFF_{16}$. Therefore, store the start address of each interrupt routine here.
The internal RAM is allocated in an upper address direction beginning with address $00400_{16}$. An internal RAM of M306H7MC-XXXFP, for instance, is allocated to the addresses from $00400_{16}$ to $017FF_{16}$. In addition to storing data, the internal RAM also stores the stack used when calling subroutines and when interrupts are generated.
SFR is allocated to the addresses from $00000_{16}$ to $003FF_{16}$. Peripheral function control registers are located here. Of SFR, any area which has no functions allocated is reserved for future use and cannot be used by users.
The special page vector table is allocated to the addresses from $FFE00_{16}$ to $FFFDB_{16}$. This vector is used by the JMPS or JSRS instruction. For details, refer to the "M16C/60 and M16C/20 Series Software Manual."
In memory expansion and microprocessor modes, some areas are reserved for future use and cannot be used by users.




| Internal RAM | | Internal ROM | |
|---|---|---|---|
| Size | Address XXXXX$_{16}$ | Size | Address YYYYY$_{16}$ |
| 5K bytes | $17FF_{16}$ | 128K bytes | $E0000_{16}$ |
| 8K bytes | $23FF_{16}$ | 256K bytes | $C0000_{16}$ |

**Figure 1.4 Memory Map**

# 2. Central Processing Unit (CPU)

Figure 2.1 shows the CPU registers. The CPU has 13 registers. Of these, R0, R1, R2, R3, A0, A1 and FB comprise a register bank. There are two register banks.

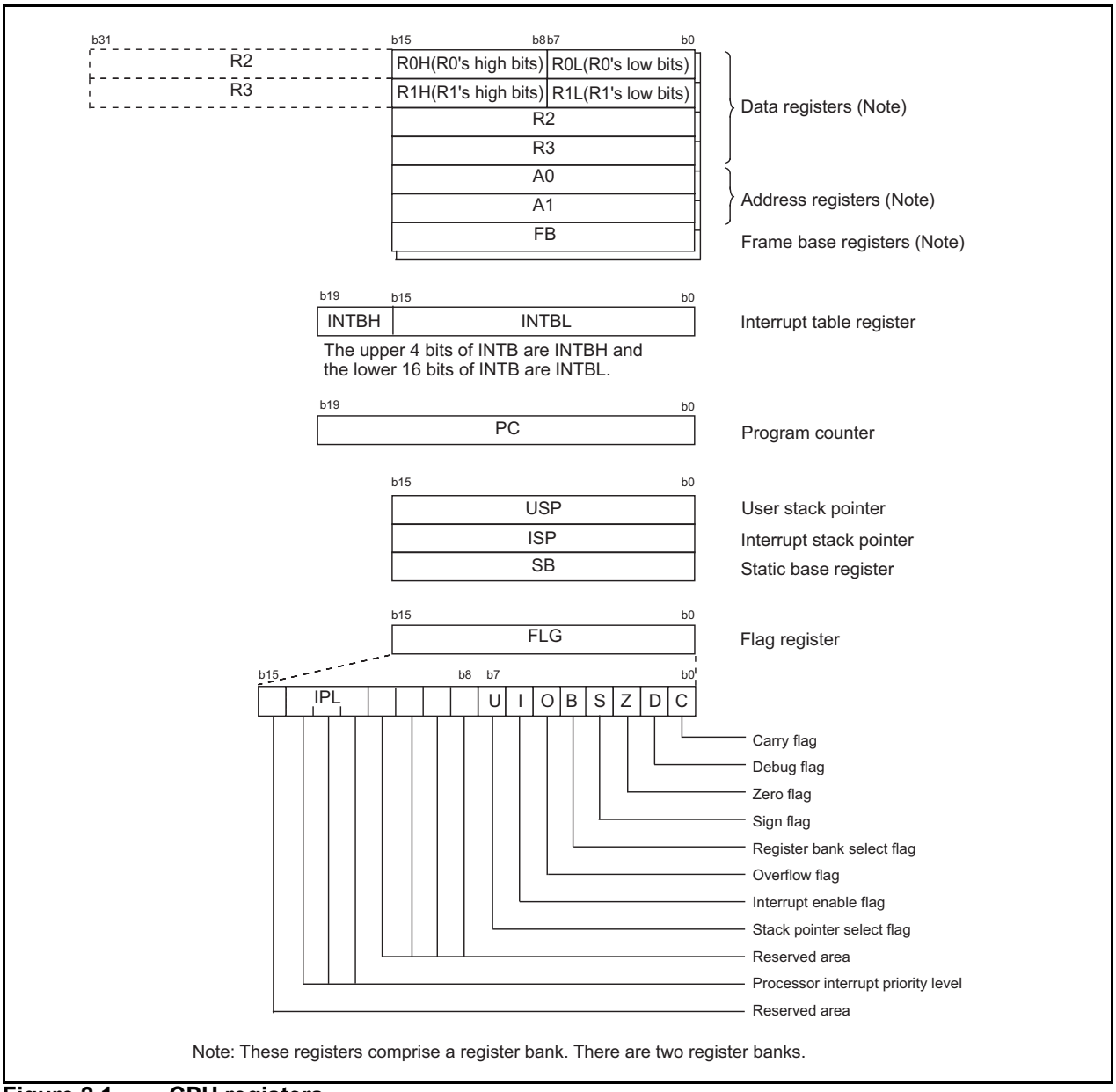



**Figure 2.1 CPU registers**

## 2.1 Data Registers (R0, R1, R2 and R3)

The R0 register consists of 16 bits, and is used mainly for transfers and arithmetic/logic operations. R1 to R3 are the same as R0.

The R0 register can be separated between high (R0H) and low (R0L) for use as two 8-bit data registers.

R1H and R1L are the same as R0H and R0L. Conversely, R2 and R0 can be combined for use as a 32- bit data register (R2R0). R3R1 is the same as R2R0.

## 2.2 Address Registers (A0 and A1)

The register A0 consists of 16 bits, and is used for address register indirect addressing and address register relative addressing. They also are used for transfers and logic/logic operations. A1 is the same as A0.

In some instructions, registers A1 and A0 can be combined for use as a 32-bit address register (A1A0).

## 2.3 Frame Base Register (FB)

FB is configured with 16 bits, and is used for FB relative addressing.

## 2.4 Interrupt Table Register (INTB)

INTB is configured with 20 bits, indicating the start address of an interrupt vector table.

## 2.5 Program Counter (PC)

PC is configured with 20 bits, indicating the address of an instruction to be executed.

## 2.6 User Stack Pointer (USP) and Interrupt Stack Pointer (ISP)

Stack pointer (SP) comes in two types: USP and ISP, each configured with 16 bits.
Your desired type of stack pointer (USP or ISP) can be selected by the U flag of FLG.

## 2.7 Static Base Register (SB)

SB is configured with 16 bits, and is used for SB relative addressing.

## 2.8 Flag Register (FLG)

FLG consists of 11 bits, indicating the CPU status.

- Carry Flag (C Flag)
  This flag retains a carry, borrow, or shift-out bit that has occurred in the arithmetic/logic unit.
- Debug Flag (D Flag)
  The D flag is used exclusively for debugging purpose. During normal use, it must be set to “0”.
- Zero Flag (Z Flag)
  This flag is set to “1” when an arithmetic operation resulted in 0; otherwise, it is “0”.
- Sign Flag (S Flag)
  This flag is set to “1” when an arithmetic operation resulted in a negative value; otherwise, it is “0”.
- Register Bank Select Flag (B Flag)
  Register bank 0 is selected when this flag is “0” ; register bank 1 is selected when this flag is “1”.
- Overflow Flag (O Flag)
  This flag is set to “1” when the operation resulted in an overflow; otherwise, it is “0”.
- Interrupt Enable Flag (I Flag)
  This flag enables a maskable interrupt.
  Maskable interrupts are disabled when the I flag is “0”, and are enabled when the I flag is “1”. The I flag is cleared to “0” when the interrupt request is accepted.
- Stack Pointer Select Flag (U Flag)
  ISP is selected when the U flag is “0”; USP is selected when the U flag is “1”.
  The U flag is cleared to “0” when a hardware interrupt request is accepted or an INT instruction for software interrupt Nos. 0 to 31 is executed.
- Processor Interrupt Priority Level (IPL)
  IPL is configured with three bits, for specification of up to eight processor interrupt priority levels from level 0 to level 7.
  If a requested interrupt has priority greater than IPL, the interrupt is enabled.
- Reserved Area
  When write to this bit, write "0". When read, its content is indeterminate.

# 3. Reset

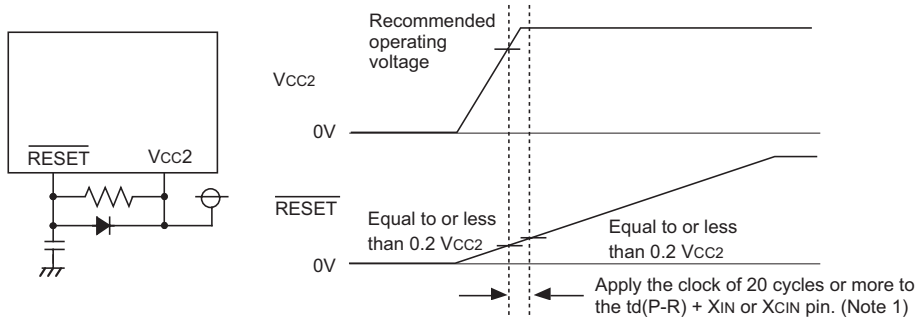
There are three types of resets: a hardware reset, a software reset, and a watchdog timer reset.

## 3.1 Hardware Reset

A reset is applied using the $\overline{\text{RESET}}$ pin. When an "L" signal is applied to the $\overline{\text{RESET}}$ pin while the power supply voltage is within the recommended operating condition, the pins are initialized (see Table 3.1).
The oscillation circuit is initialized and the main clock starts oscillating. When the input level at the $\overline{\text{RESET}}$ pin is released from "L" to "H", the CPU and SFR are initialized, and the program is executed starting from the address indicated by the reset vector. The internal RAM is not initialized. If the $\overline{\text{RESET}}$ pin is pulled "L" while writing to the internal RAM, the internal RAM becomes indeterminate.
Figure 3.1 shows the example reset circuit. Figure 3.2 shows the reset sequence. Table 3.1 shows the statuses of the other pins while the $\overline{\text{RESET}}$ pin is "L". Figure 3.3 shows the CPU register status after reset.
Refer to "SFR" for SFR status after reset.

1. When the power supply is stable
   - When STARTB pin = "L"
     (1) Apply an "L" signal to the $\overline{\text{RESET}}$ pin.
     (2) Apply a clock for 20 cycles or more to the XIN pin.
     (3) Apply an "H" signal to the $\overline{\text{RESET}}$ pin.
   - When STARTB pin = "H"
     (1) Apply an "L" signal to the $\overline{\text{RESET}}$ pin.
     (2) Apply a clock for 20 cycles or more to the XCIN pin.
     (3) Apply an "H" signal to the $\overline{\text{RESET}}$ pin.

2. Power on
   - When STARTB pin = "L"
     (1) Apply an "L" signal to the $\overline{\text{RESET}}$ pin.
     (2) Let the power supply voltage increase until it meets the recommended operating condition.
     (3) Wait td(P-R) or more until the internal power supply is stabilized.
     (4) Apply a clock for 20 cycles or more to the XIN pin.
     (5) Apply an "H" signal to the $\overline{\text{RESET}}$ pin.
   - When STARTB pin = "H"
     (1) Apply an "L" signal to the $\overline{\text{RESET}}$ pin.
     (2) Let the power supply voltage increase until it meets the recommended operating condition.
     (3) Wait td(P-R) or more until the internal power supply is stabilized.
     (4) Apply a clock for 20 cycles or more to the XCIN pin.
     (5) Apply an "H" signal to the $\overline{\text{RESET}}$ pin.




Notes 1: When the STARTB pin=L, apply the clock of 20 cycles or more to the XIN pin.
When the STARTB pin=H, apply the clock of 20 cycles or more to the XCIN pin.
Notes 2: If $V_{CC1} \leq V_{CC2}$, the $V_{CC1}$ voltage must be lower than that of $V_{CC2}$ when the power is being turned on or off.

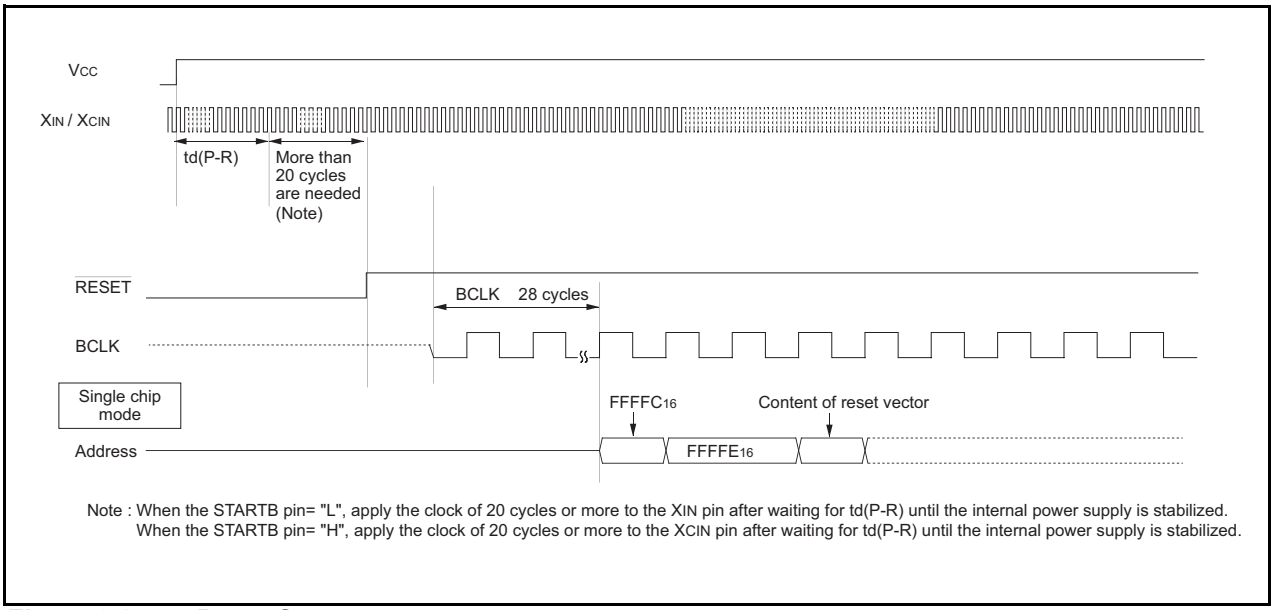
**Figure 3.1 Example Reset Circuit**

## 3.2 Software Reset

When the PM03 bit in the PM0 register is set to "1" (microcomputer reset), the microcomputer has its pins, CPU, and SFR initialized. Then the program is executed starting from the address indicated by the reset vector.
Select the main clock for the CPU clock source, and set the PM03 bit to "1" with main clock oscillation satisfactorily stable.
At software reset, some SFR's are not initialized. Refer to "SFR". Also, since the PM01 to PM00 bits in the PM0 register are not initialized, the processor mode remains unchanged.

## 3.3 Watchdog Timer Reset

Where the PM12 bit in the PM1 register is "1" (reset when watchdog timer underflows), the microcomputer initializes its pins, CPU and SFR if the watchdog timer underflows. Then the program is executed starting from the address indicated by the reset vector.
At watchdog timer reset, some SFR's are not initialized. Refer to "SFR". Also, since the PM01 to PM00 bits in the PM0 register are not initialized, the processor mode remains unchanged.




Note : When the STARTB pin= "L", apply the clock of 20 cycles or more to the XIN pin after waiting for td(P-R) until the internal power supply is stabilized.
When the STARTB pin= "H", apply the clock of 20 cycles or more to the XCIN pin after waiting for td(P-R) until the internal power supply is stabilized.

**Figure 3.2 Reset Sequence**

**Table 3.1 Pin Status When RESET Pin Level is "L**

| Pin name | Status | | |
|---|---|---|---|
| | CNVSS = VSS | CNVSS = VCC (Note) | |
| | | BYTE = VSS | BYTE = VCC |
| P0 | Input port | Data input | Data input |
| P1 | Input port | Data input | Input port |
| P2, P3, P40 to P43 | Input port | Address output (undefined) | Address output (undefined) |
| P44 | Input port | $\overline{CS0}$ output ("H" is output) | $\overline{CS0}$ output ("H" is output) |
| P45 to P47 | Input port | Input port (Pulled high) | Input port (Pulled high) |
| P50 | Input port | $\overline{WR}$ output ("H" is output) | $\overline{WR}$ output ("H" is output) |
| P51 | Input port | $\overline{BHE}$ output (undefined) | $\overline{BHE}$ output (undefined) |
| P52 | Input port | $\overline{RD}$ output ("H" is output) | $\overline{RD}$ output ("H" is output) |
| P53 | Input port | BCLK output | BCLK output |
| P54 | Input port | $\overline{HLDA}$ output (The output value depends on the input to the $\overline{HOLD}$ pin) | $\overline{HLDA}$ output (The output value depends on the input to the $\overline{HOLD}$ pin) |
| P55 | Input port | $\overline{HOLD}$ input | $\overline{HOLD}$ input |
| P56 | Input port | ALE output ("L" is output) | ALE output ("L" is output) |
| P57 | Input port | $\overline{RDY}$ input | $\overline{RDY}$ input |
| P6, P7, P80 to P84, P86, P87, P9 | Input port | Input port | Input port |

Note : Do not set CNVss=Vcc for this product.

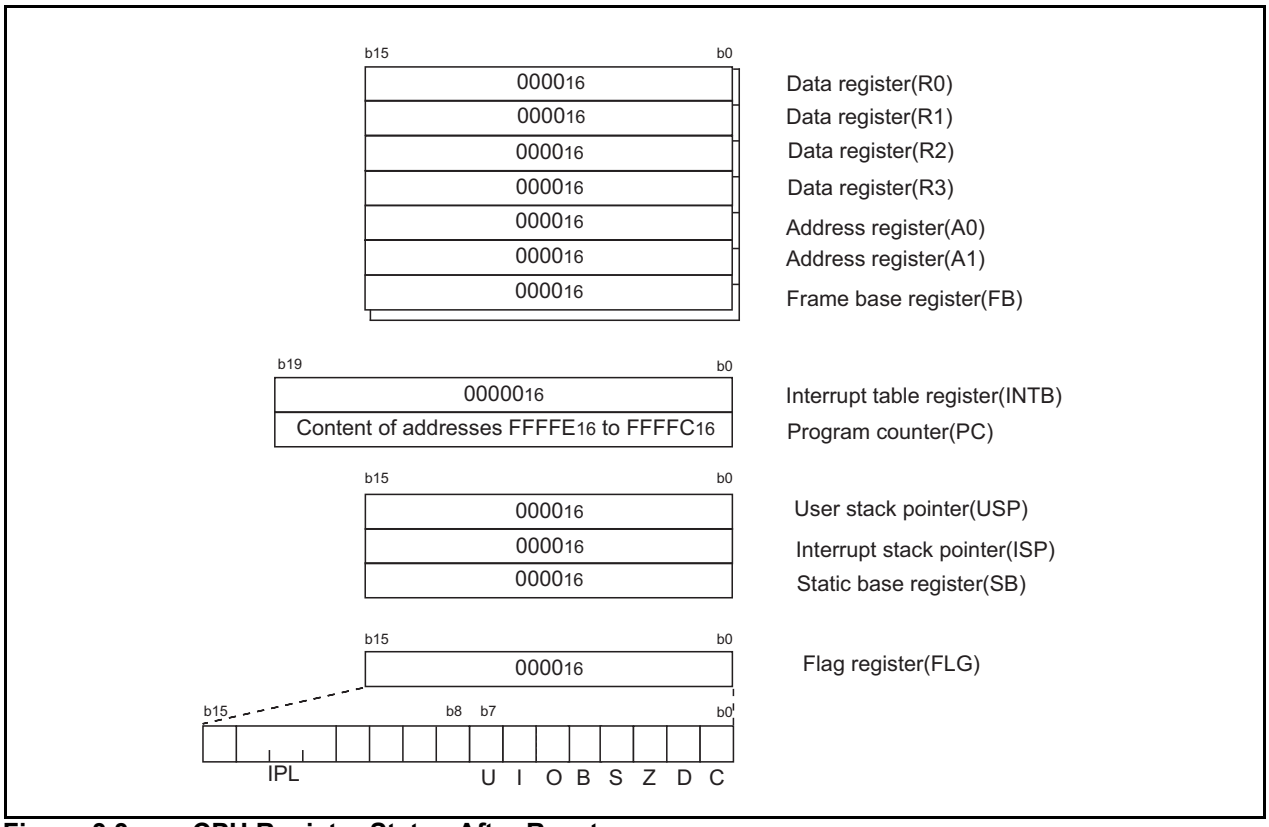



**Figure 3.3 CPU Register Status After Reset**

## 3.4 SFR

| Address | Register (Note 1) | Symbol | After reset |
|---|---|---|---|
| 0000₁₆ | | | |
| 0001₁₆ | | | |
| 0002₁₆ | | | |
| 0003₁₆ | | | |
| 0004₁₆ | Processor mode register 0 (Note 2) | PM0 | 00000000₂ |
| 0005₁₆ | Processor mode register 1 | PM1 | 00001000₂ |
| 0006₁₆ | System clock control register 0 | CM0 | 01001000₂(the STARTB pin is "L")<br>01111000₂(the STARTB pin is "H") |
| 0007₁₆ | System clock control register 1 | CM1 | 00100000₂ |
| 0008₁₆ | | | |
| 0009₁₆ | Address match interrupt enable register | AIER | XXXXXX00₂ |
| 000A₁₆ | Protect register | PRCR | XX000000₂ |
| 000B₁₆ | | | |
| 000C₁₆ | | | |
| 000D₁₆ | | | |
| 000E₁₆ | Watchdog timer start register | WDTS | XX₁₆ |
| 000F₁₆ | Watchdog timer control register | WDC | 00XXXXXX₂(Note 3) |
| 0010₁₆<br>0011₁₆<br>0012₁₆ | Address match interrupt register 0 | RMAD0 | 00₁₆<br>00₁₆<br>X0₁₆ |
| 0013₁₆ | | | |
| 0014₁₆<br>0015₁₆<br>0016₁₆ | Address match interrupt register 1 | RMAD1 | 00₁₆<br>00₁₆<br>X0₁₆ |
| 0017₁₆ | | | |
| 0018₁₆ | | | |
| 0019₁₆ | | | |
| 001A₁₆ | | | |
| 001B₁₆ | | | |
| 001C₁₆ | | | |
| 001D₁₆ | | | |
| 001E₁₆ | Processor mode register 2 | PM2 | XXX00000₂ |
| 001F₁₆ | | | |
| 0020₁₆<br>0021₁₆<br>0022₁₆ | DMA0 source pointer | SAR0 | XX₁₆<br>XX₁₆<br>XX₁₆ |
| 0023₁₆ | | | |
| 0024₁₆<br>0025₁₆<br>0026₁₆ | DMA0 destination pointer | DAR0 | XX₁₆<br>XX₁₆<br>XX₁₆ |
| 0027₁₆ | | | |
| 0028₁₆<br>0029₁₆ | DMA0 transfer counter | TCR0 | XX₁₆<br>XX₁₆ |
| 002A₁₆ | | | |
| 002B₁₆ | | | |
| 002C₁₆ | DMA0 control register | DM0CON | 00000X00₂ |
| 002D₁₆ | | | |
| 002E₁₆ | | | |
| 002F₁₆ | | | |
| 0030₁₆<br>0031₁₆<br>0032₁₆ | DMA1 source pointer | SAR1 | XX₁₆<br>XX₁₆<br>XX₁₆ |
| 0033₁₆ | | | |
| 0034₁₆<br>0035₁₆<br>0036₁₆ | DMA1 destination pointer | DAR1 | XX₁₆<br>XX₁₆<br>XX₁₆ |
| 0037₁₆ | | | |
| 0038₁₆<br>0039₁₆ | DMA1 transfer counter | TCR1 | XX₁₆<br>XX₁₆ |
| 003A₁₆ | | | |
| 003B₁₆ | | | |
| 003C₁₆ | DMA1 control register | DM1CON | 00000X00₂ |
| 003D₁₆ | | | |
| 003E₁₆ | | | |
| 003F₁₆ | | | |

Note 1: The blank areas are reserved and cannot be accessed by users.
Note 2: The PM00 and PM01 bits do not change at software reset, watchdog timer reset and oscillation stop detection reset.
Note 3: The WDC5 bit is "0" (cold start) immediately after power-on. It can only be set to "1" in a program.

X : Undefined

| Address | Register | Symbol | After reset |
|---|---|---|---|
| 0040₁₆ | | | |
| 0041₁₆ | | | |
| 0042₁₆ | | | |
| 0043₁₆ | | | |
| 0044₁₆ | INT3 interrupt control register | INT3IC | XX00X0002 |
| 0045₁₆ | Timer B5/SLICE ON interrupt control register | TB5IC | XXXXX0002 |
| 0046₁₆ | Timer B4/Remote control interrupt control register, UART1 BUS collision detection interrupt control register | TB4IC, U1BCNIC | XXXXX0002 |
| 0047₁₆ | Timer B3/HINT interrupt control register, UART0 BUS collision detection interrupt control register | TB3IC, U0BCNIC | XXXXX0002 |
| 0048₁₆ | SI/O4 interrupt control register, INT5 interrupt control register | S4IC, INT5IC | XX00X0002 |
| 0049₁₆ | SI/O3 interrupt control register, INT4 interrupt control register | S3IC, INT4IC | XX00X0002 |
| 004A₁₆ | UART2 Bus collision detection interrupt control register | BCNIC | XXXXX0002 |
| 004B₁₆ | DMA0 interrupt control register | DM0IC | XXXXX0002 |
| 004C₁₆ | DMA1 interrupt control register | DM1IC | XXXXX0002 |
| 004D₁₆ | | | |
| 004E₁₆ | A/D conversion interrupt control register | ADIC | XXXXX0002 |
| 004F₁₆ | UART2 transmit interrupt control register | S2TIC | XXXXX0002 |
| 0050₁₆ | UART2 receive interrupt control register | S2RIC | XXXXX0002 |
| 0051₁₆ | UART0 transmit interrupt control register | S0TIC | XXXXX0002 |
| 0052₁₆ | UART0 receive interrupt control register | S0RIC | XXXXX0002 |
| 0053₁₆ | UART1 transmit interrupt control register | S1TIC | XXXXX0002 |
| 0054₁₆ | UART1 receive interrupt control register | S1RIC | XXXXX0002 |
| 0055₁₆ | Timer A0 interrupt control register | TA0IC | XXXXX0002 |
| 0056₁₆ | Timer A1 interrupt control register | TA1IC | XXXXX0002 |
| 0057₁₆ | Timer A2 interrupt control register | TA2IC | XXXXX0002 |
| 0058₁₆ | Timer A3 interrupt control register | TA3IC | XXXXX0002 |
| 0059₁₆ | Timer A4 interrupt control register | TA4IC | XXXXX0002 |
| 005A₁₆ | Timer B0 interrupt control register | TB0IC | XXXXX0002 |
| 005B₁₆ | Timer B1 interrupt control register | TB1IC | XXXXX0002 |
| 005C₁₆ | Timer B2/Clock timer interrupt control register | TB2IC | XXXXX0002 |
| 005D₁₆ | INT0 interrupt control register | INT0IC | XX00X0002 |
| 005E₁₆ | INT1 interrupt control register | INT1IC | XX00X0002 |
| 005F₁₆ | INT2 interrupt control register | INT2IC | XX00X0002 |
| 0060₁₆ | | | |
| 0061₁₆ | | | |
| 0062₁₆ | | | |
| 0063₁₆ | | | |
| 0064₁₆ | | | |
| 0065₁₆ | | | |
| 0066₁₆ | | | |
| 0067₁₆ | | | |
| 0068₁₆ | | | |
| 0069₁₆ | | | |
| 006A₁₆ | | | |
| 006B₁₆ | | | |
| 006C₁₆ | | | |
| 006D₁₆ | | | |
| 006E₁₆ | | | |
| 006F₁₆ | | | |
| 0070₁₆ | | | |
| 0071₁₆ | | | |
| 0072₁₆ | | | |
| 0073₁₆ | | | |
| 0074₁₆ | | | |
| 0075₁₆ | | | |
| 0076₁₆ | | | |
| 0077₁₆ | | | |
| 0078₁₆ | | | |
| 0079₁₆ | | | |
| 007A₁₆ | | | |
| 007B₁₆ | | | |
| 007C₁₆ | | | |
| 007D₁₆ | | | |
| 007E₁₆ | | | |
| 007F₁₆ | | | |

Note :The blank areas are reserved and cannot be accessed by users.

X : Undefined

| Address | Register | Symbol | After reset |
|---|---|---|---|
| 008016 | | | |
| 008116 | | | |
| 008216 | | | |
| 008316 | | | |
| 008416 | | | |
| 008516 | | | |
| 008616 | | | |
| ≈ | | | ≈ |
| 01B016 | | | |
| 01B116 | | | |
| 01B216 | | | |
| 01B316 | | | |
| 01B416 | | | |
| 01B516 | Flash memory control register 1 (Note 2) | FMR1 | 0X00XX0X2 |
| 01B616 | | | |
| 01B716 | Flash memory control register 0 (Note 2) | FMR0 | XX000001₂ |
| 01B816<br>01B916<br>01BA16 | Address match interrupt register 2 | RMAD2 | 0016<br>0016<br>X016 |
| 01BB16 | Address match interrupt enable register 2 | AIER2 | XXXXXX002 |
| 01BC16<br>01BD16<br>01BE16 | Address match interrupt register 3 | RMAD3 | 0016<br>0016<br>X016 |
| 01BF16 | | | |
| ≈ | | | ≈ |
| 020016<br>020116 | Remote control transmission buffer register | RMTTMHL | 0016<br>0016 |
| ≈ | | | ≈ |
| 020E16<br>020F16 | Slice RAM address control register | SA | 0016 |
| 021016<br>021116 | Slice RAM data control register | SD | 0016 |
| 021216<br>021316 | Address control register for CRC registers | CA | 0016 |
| 021416<br>021516 | Data control register for CRC registers | CD | 0016 |
| 021616<br>021716 | Address control register for extended registers | DA | 0016 |
| 021816<br>021916 | Data control register for extended registers | DD | 0016 |
| 021A16<br>021B16 | Humming 8/4 register | HM8 | 0016 |
| 021C16<br>021D16 | Humming 24/18 register 0 | HM0 | 0016 |
| 021E16<br>021F16 | Humming 24/18 register 1 | HM1 | 0016 |
| 025016 | | | |
| ≈ | | | ≈ |
| 025916 | | | |
| 025A16 | | | |
| 025B16 | | | |
| 025C16 | | | |
| 025D16 | | | |
| 025E16 | Peripheral clock select register | PCLKR | 000000112 |
| 025F16 | | | |
| ≈ | | | ≈ |
| 02D616 | I2C0 interrupt control register | EXTIICINT | 0016 |
| 02D716 | Reserved register | EXTREG02D7 | 0016 |
| ≈ | | | |
| 02E016 | I2C data shift register | IIC0S0 | Undefined |
| 02E116 | I2C address register | IIC0S0D | 0016 |
| 02E216 | I2C states register | IIC0S1 | 0001000?2 |
| 02E316 | I2C control register | IIC0S1D | 0016 |
| 02E416 | I2C clock control register | IIC0S2 | 0016 |
| 02E516 | Reserved register | REVREG02E5 | 00?000002 |
| 02E616 | I2C transmit buffer register | IIC0S0S | Undefined |
| ≈ | | | |
| 033016 | | | |
| 033116 | | | |
| 033216 | | | |
| 033316 | | | |

Note 1: The blank areas are reserved and cannot be accessed by users.
Note 2: This register is included in the flash memory version.

X : Undefined

| Address | Register | Symbol | After reset |
|---|---|---|---|
| 0340₁₆ | Timer B3, 4, 5 count start flag | TBSR | 000XXXXX₂ |
| 0341₁₆ | | | |
| 0342₁₆<br>0343₁₆ | | | |
| 0344₁₆<br>0345₁₆ | | | |
| 0346₁₆<br>0347₁₆ | | | |
| 0348₁₆ | | | |
| 0349₁₆ | | | |
| 034A₁₆ | | | |
| 034B₁₆ | | | |
| 034C₁₆ | | | |
| 034D₁₆ | | | |
| 034E₁₆ | | | |
| 034F₁₆ | | | |
| 0350₁₆<br>0351₁₆ | Timer B3 register | TB3 | XX₁₆<br>XX₁₆ |
| 0352₁₆<br>0353₁₆ | Timer B4 register | TB4 | XX₁₆<br>XX₁₆ |
| 0354₁₆<br>0355₁₆ | Timer B5 register | TB5 | XX₁₆<br>XX₁₆ |
| 0356₁₆ | | | |
| 0357₁₆ | | | |
| 0358₁₆ | | | |
| 0359₁₆ | | | |
| 035A₁₆ | | | |
| 035B₁₆ | Timer B3 mode register | TB3MR | 00XX0000₂ |
| 035C₁₆ | Timer B4 mode register | TB4MR | 00XX0000₂ |
| 035D₁₆ | Timer B5 mode register | TB5MR | 00XX0000₂ |
| 035E₁₆ | Interrupt cause select register 2 | IFSR2A | 00XXXXXX₂ |
| 035F₁₆ | Interrupt cause select register | IFSR | 00₁₆ |
| 0360₁₆ | SI/O3 transmit/receive register | S3TRR | XX₁₆ |
| 0361₁₆ | | | |
| 0362₁₆ | SI/O3 control register | S3C | 01000000₂ |
| 0363₁₆ | SI/O3 bit rate generator register | S3BRG | XX₁₆ |
| 0364₁₆ | SI/O4 transmit/receive register | S4TRR | XX₁₆ |
| 0365₁₆ | | | |
| 0366₁₆ | SI/O4 control register | S4C | 01000000₂ |
| 0367₁₆ | SI/O4 bit rate generator register | S4BRG | XX₁₆ |
| 0368₁₆ | | | |
| 0369₁₆ | | | |
| 036A₁₆ | | | |
| 036B₁₆ | | | |
| 036C₁₆ | UART0 special mode register 4 | U0SMR4 | 00₁₆ |
| 036D₁₆ | UART0 special mode register 3 | U0SMR3 | 000X0X0X₂ |
| 036E₁₆ | UART0 special mode register 2 | U0SMR2 | X0000000₂ |
| 036F₁₆ | UART0 special mode register | U0SMR | X0000000₂ |
| 0370₁₆ | UART1 special mode register 4 | U1SMR4 | 00₁₆ |
| 0371₁₆ | UART1 special mode register 3 | U1SMR3 | 000X0X0X₂ |
| 0372₁₆ | UART1 special mode register 2 | U1SMR2 | X0000000₂ |
| 0373₁₆ | UART1 special mode register | U1SMR | X0000000₂ |
| 0374₁₆ | UART2 special mode register 4 | U2SMR4 | 00₁₆ |
| 0375₁₆ | UART2 special mode register 3 | U2SMR3 | 000X0X0X₂ |
| 0376₁₆ | UART2 special mode register 2 | U2SMR2 | X0000000₂ |
| 0377₁₆ | UART2 special mode register | U2SMR | X0000000₂ |
| 0378₁₆ | UART2 transmit/receive mode register | U2MR | 00₁₆ |
| 0379₁₆ | UART2 bit rate generator | U2BRG | XX₁₆ |
| 037A₁₆<br>037B₁₆ | UART2 transmit buffer register | U2TB | XXXXXXXX₂<br>XXXXXXXX₂ |
| 037C₁₆ | UART2 transmit/receive control register 0 | U2C0 | 00001000₂ |
| 037D₁₆ | UART2 transmit/receive control register 1 | U2C1 | 00000010₂ |
| 037E₁₆<br>037F₁₆ | UART2 receive buffer register | U2RB | XXXXXXXX₂<br>XXXXXXXX₂ |

Note : The blank areas are reserved and cannot be accessed by users.
X : Undefined

| Address | Register | Symbol | After reset |
|---|---|---|---|
| 038016 | Count start flag | TABSR | 0016 |
| 038116 | Clock prescaler reset flag | CPSRF | 0XXXXXXX2 |
| 038216 | One-shot start flag | ONSF | 0016 |
| 038316 | Trigger select register | TRGSR | 0016 |
| 038416 | Up-down flag | UDF | 0016 |
| 038516 | | | |
| 038616 | Timer A0 register | TA0 | XX16 |
| 038716 | | | XX16 |
| 038816 | Timer A1 register | TA1 | XX16 |
| 038916 | | | XX16 |
| 038A16 | Timer A2 register | TA2 | XX16 |
| 038B16 | | | XX16 |
| 038C16 | Timer A3 register | TA3 | XX16 |
| 038D16 | | | XX16 |
| 038E16 | Timer A4 register | TA4 | XX16 |
| 038F16 | | | XX16 |
| 039016 | Timer B0 register | TB0 | XX16 |
| 039116 | | | XX16 |
| 039216 | Timer B1 register | TB1 | XX16 |
| 039316 | | | XX16 |
| 039416 | Timer B2 register | TB2 | XX16 |
| 039516 | | | XX16 |
| 039616 | Timer A0 mode register | TA0MR | 0016 |
| 039716 | Timer A1 mode register | TA1MR | 0016 |
| 039816 | Timer A2 mode register | TA2MR | 0016 |
| 039916 | Timer A3 mode register | TA3MR | 0016 |
| 039A16 | Timer A4 mode register | TA4MR | 0016 |
| 039B16 | Timer B0 mode register | TB0MR | 00XX00002 |
| 039C16 | Timer B1 mode register | TB1MR | 00XX00002 |
| 039D16 | Timer B2 mode register | TB2MR | 00XX00002 |
| 039E16 | | | |
| 039F16 | | | |
| 03A016 | UART0 transmit/receive mode register | U0MR | 0016 |
| 03A116 | UART0 bit rate generator register | U0BRG | XX16 |
| 03A216 | UART0 transmit buffer register | U0TB | XXXXXXXX2 |
| 03A316 | | | XXXXXXXX2 |
| 03A416 | UART0 transmit/receive control register 0 | U0C0 | 000010002 |
| 03A516 | UART0 transmit/receive control register 1 | U0C1 | 000000102 |
| 03A616 | UART0 receive buffer register | U0RB | XXXXXXXX2 |
| 03A716 | | | XXXXXXXX2 |
| 03A816 | UART1 transmit/receive mode register | U1MR | 0016 |
| 03A916 | UART1 bit rate generator | U1BRG | XX16 |
| 03AA16 | UART1 transmit buffer register | U1TB | XXXXXXXX2 |
| 03AB16 | | | XXXXXXXX2 |
| 03AC16 | UART1 transmit/receive control register 0 | U1C0 | 000010002 |
| 03AD16 | UART1 transmit/receive control register 1 | U1C1 | 000000102 |
| 03AE16 | UART1 receive buffer register | U1RB | XXXXXXXX2 |
| 03AF16 | | | XXXXXXXX2 |
| 03B016 | UART transmit/receive control register 2 | UCON | X00000002 |
| 03B116 | | | |
| 03B216 | | | |
| 03B316 | | | |
| 03B416 | | | |
| 03B516 | | | |
| 03B616 | | | |
| 03B716 | | | |
| 03B816 | DMA0 request cause select register | DM0SL | 0016 |
| 03B916 | | | |
| 03BA16 | DMA1 request cause select register | DM1SL | 0016 |
| 03BB16 | | | |
| 03BC16 | CRC data register | CRCD | XX16 |
| 03BD16 | | | XX16 |
| 03BE16 | CRC input register | CRCIN | XX16 |
| 03BF16 | | | |

Note : The blank areas are reserved and cannot be accessed by users.
X : Undefined

| Address | Register | Symbol | After reset |
|---|---|---|---|
| 03C016 | A/D register 0 | AD0 | XXXXXXXX2 |
| 03C116 | | | |
| 03C216 | A/D register 1 | AD1 | XXXXXXXX2 |
| 03C316 | | | |
| 03C416 | A/D register 2 | AD2 | XXXXXXXX2 |
| 03C516 | | | |
| 03C616 | A/D register 3 | AD3 | XXXXXXXX2 |
| 03C716 | | | |
| 03C816 | A/D register 4 | AD4 | XXXXXXXX2 |
| 03C916 | | | |
| 03CA16 | A/D register 5 | AD5 | XXXXXXXX2 |
| 03CB16 | | | |
| 03CC16 | A/D register 6 | AD6 | XXXXXXXX2 |
| 03CD16 | | | |
| 03CE16 | A/D register 7 | AD7 | XXXXXXXX2 |
| 03CF16 | | | |
| 03D016 | | | |
| 03D116 | | | |
| 03D216 | | | |
| 03D316 | | | |
| 03D416 | A/D control register 2 | ADCON2 | 0016 |
| 03D516 | | | |
| 03D616 | A/D control register 0 | ADCON0 | 00000XXX2 |
| 03D716 | A/D control register 1 | ADCON1 | 0016 |
| 03D816 | | | |
| 03D916 | | | |
| 03DA16 | | | |
| 03DB16 | | | |
| 03DC16 | | | |
| 03DD16 | | | |
| 03DE16 | | | |
| 03DF16 | | | |
| 03E016 | Port P0 register | P0 | XX16 |
| 03E116 | Port P1 register | P1 | XX16 |
| 03E216 | Port P0 direction register | PD0 | 0016 |
| 03E316 | Port P1 direction register | PD1 | 0016 |
| 03E416 | Port P2 register | P2 | XX16 |
| 03E516 | Port P3 register | P3 | XX16 |
| 03E616 | Port P2 direction register | PD2 | 0016 |
| 03E716 | Port P3 direction register | PD3 | 0016 |
| 03E816 | Port P4 register | P4 | XX16 |
| 03E916 | Port P5 register | P5 | XX16 |
| 03EA16 | Port P4 direction register | PD4 | 0016 |
| 03EB16 | Port P5 direction register | PD5 | 0016 |
| 03EC16 | Port P6 register | P6 | XX16 |
| 03ED16 | Port P7 register | P7 | XX16 |
| 03EE16 | Port P6 direction register | PD6 | 0016 |
| 03EF16 | Port P7 direction register | PD7 | 0016 |
| 03F016 | Port P8 register | P8 | XX16 |
| 03F116 | Port P9 register | P9 | XX16 |
| 03F216 | Port P8 direction register | PD8 | 00X000002 |
| 03F316 | Port P9 direction register | PD9 | 0016 |
| 03F416 | Port P10 register | P10 | XX16 |
| 03F516 | | | |
| 03F616 | | | |
| 03F716 | | | |
| 03F816 | | | |
| 03F916 | | | |
| 03FA16 | | | |
| 03FB16 | | | |
| 03FC16 | Pull-up control register 0 | PUR0 | 0016 |
| 03FD16 | Pull-up control register 1 | PUR1 | 000000002 |
| 03FE16 | Pull-up control register 2 | PUR2 | 0016 |
| 03FF16 | Port control register | PCR | 0016 |

Note 1: The blank areas are reserved and cannot be accessed by users.

Processor mode register 0 (Note 1)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | | 0 | | |

| Symbol | Address | After reset |
|---|---|---|
| PM0 | $0004_{16}$ | $00000000_2$ (CNVSS pin = "L") |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| PM00 | Processor mode bit | b1 b0<br>0 0: Single-chip mode<br>0 1: Memory expansion mode<br>1 0: Must not be set<br>1 1: Microprocessor mode | RW |
| PM01 | | | RW |
| (b2) | Reserved bit | Must set to "0". | RW |
| PM03 | Software reset bit | Setting this bit to “1” resets the microcomputer. When read, its content is "0". | RW |
| (b7-b4) | Reserved bits | Must set to "0". | RW |

Note 1: Write to this register after setting the PRC1 bit in the PRCR register to "1" (write enable).

**Figure 3.4 PM0 Register**

Processor mode register 1 (Note 1)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | 0 | 0 | 0 | 0 | | 0 | 0 |

Symbol: PM1
Address: 0005$_{16}$
After reset: 0X001000$_2$

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| (b1-b0) | Reserved bit | Must set to "0". | RW |
| PM12 | Watchdog timer function select bit | 0 : Watchdog timer interrupt<br>1 : Watchdog timer reset (Note 2) | RW |
| (b6-b3) | Reserved bit | Must set to "0". | RW |
| PM17 | Wait bit (Note 3) | 0 : No wait state<br>1 : With wait state (1 wait) | RW |

Note 1: Write to this register after setting the PRC1 bit in the PRCR register to “1” (write enable).
Note 2: PM12 bit is set to “1” by writing a “1” in a program. (Writing a “0” has no effect.)
Note 3: When PM17 bit is set to “1” (with wait state), one wait state is inserted when accessing internal RAM or internal ROM.

**Figure 3.5 PM1 Register**

# 4. Clock Generation Circuit

The clock generation circuit contains two oscillator circuits as follows:

**(1) Main clock oscillation circuit**

**(2) Sub clock oscillation circuit**

Table 4.1 lists the clock generation circuit specifications. Figure 4.1 shows the clock generation circuit. Figures 4.2 to 4.4 show the clock-related registers.

**Table 4.1 CPU registers**

| Item | Main clock oscillation circuit | Sub clock oscillation circuit |
|---|---|---|
| Use of clock | • CPU clock source<br>• Peripheral function clock source | •CPU clock source<br>• Timer A, B's clock source |
| Clock frequency | 0 to 16 MHz (Note 3) | 32.768 kHz |
| Usable oscillator | • Ceramic oscillator<br>• Crystal oscillator (Note 2) | • Crystal oscillator |
| Pins to connect oscillator | XIN, XOUT | XCIN, XCOUT |
| Oscillation stop, restart function | Presence | Presence |
| Oscillator status after reset (Note1) | Oscillating | Stopped |
| Other | Externally derived clock can be input | |

Note 1. The state that the STARTB pin is held "L" after reset is shown.
The state that the STARTB pin is held "H" after reset is following.
Main clock oscillation circuit: Stopped
Sub clock oscillation circuit: Oscillating

Note 2. If you use "14 Expansion Function (Data acquisition)", be sure to connect a crystal oscillator between the XIN and XOUT pins.

Note 3. If you use "14 Expansion Function (Data acquisition)", connect a crystal of 10MHz, 12MHz, 14MHz, or 16MHz.

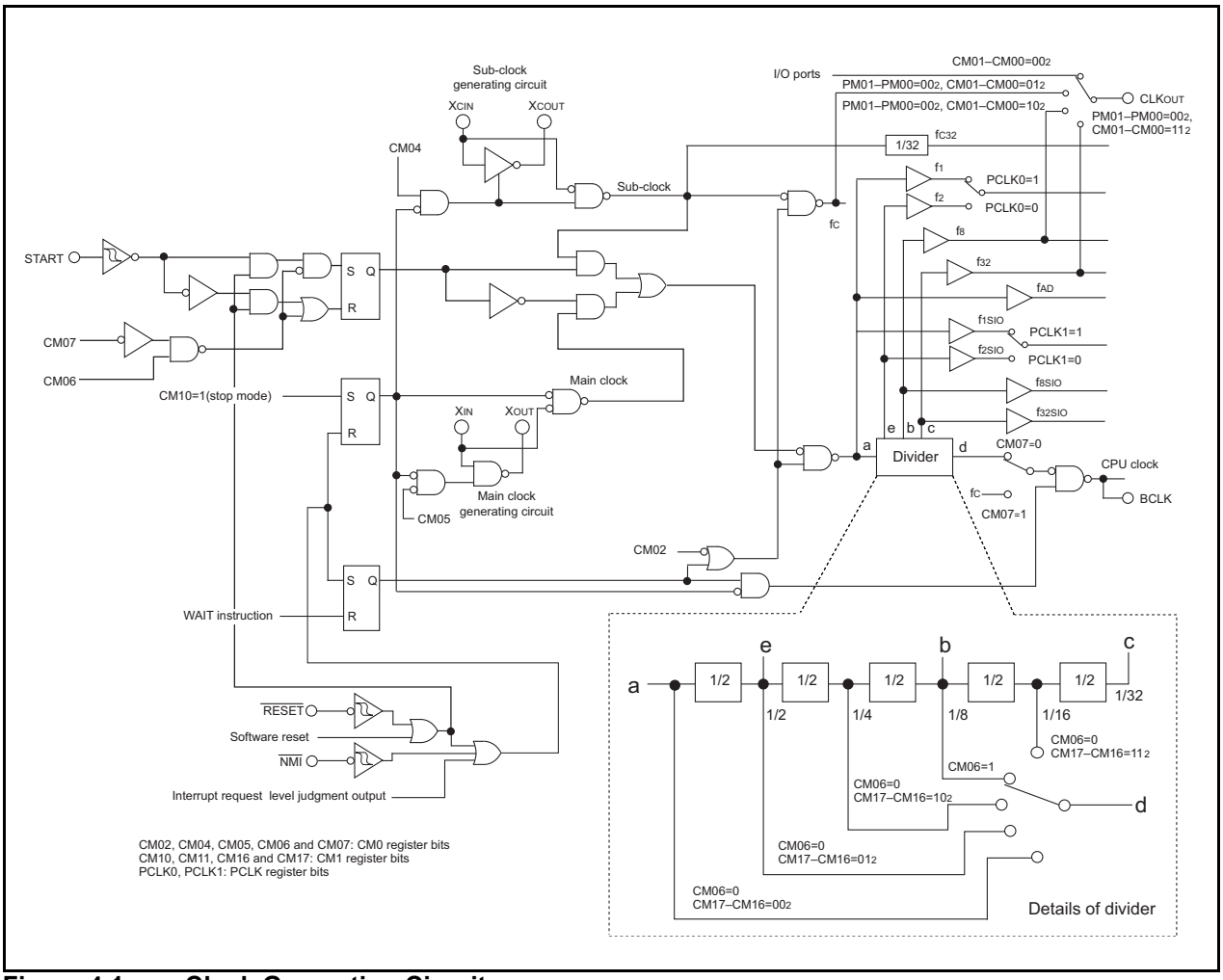
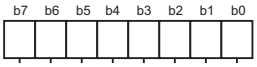



**Figure 4.1 Clock Generation Circuit**

System clock control register 0 (Note 1)




Symbol: CM0 | Address: 000616 | After reset (Note 14): 011110002 (STARTB pin = Vcc), 010010002 (STARTB pin = Vss)

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| CM00 | Clock output function select bit (Valid only in single-chip mode) | b1 b0<br>0 0 : I/O port P57<br>0 1 : fC output<br>1 0 : f8 output<br>1 1 : f32 output | RW |
| CM01 | | | RW |
| CM02 | WAIT peripheral function clock stop bit (Note 10) | 0 : Do not stop peripheral function clock in wait mode<br>1 : Stop peripheral function clock in wait mode (Note 8) | RW |
| CM03 | XCIN-XCOUT drive capacity select bit (Note 2) | 0 : LOW<br>1 : HIGH | RW |
| CM04 | Port XC select bit (Note 2) | 0 : I/O port P86, P87<br>1 : XCIN-XCOUT generation function(Note 9) | RW |
| CM05 | Main clock stop bit (Notes 3, 10, 12, 13) | 0 : On<br>1 : Off (Note 4, Note5) | RW |
| CM06 | Main clock division select bit 0 (Notes 7, 13) | 0 : CM16 and CM17 valid<br>1 : Division by 8 mode | RW |
| CM07 | System clock select bit (Notes 6, 10, 11, 12) | 0 : Main clock<br>1 : Sub-clock | RW |

Note 1: Write to this register after setting the PRC0 bit of PRCR register to “1” (write enable).
Note 2: The CM03 bit is set to “1” (high) when the CM04 bit is set to “0” (I/O port) or the microcomputer goes to a stop mode.
Note 3: This bit is provided to stop the main clock when the low power dissipation mode is selected. This bit cannot be used for detection as to whether the main clock stopped or not. To stop the main clock, the following setting is required:
(1) Set the CM07 bit to “1” (Sub-clock select) with the sub-clock stably oscillating.
(2) Set the CM05 bit to “1” (Stop).
Note 4: During external clock input, only the clock oscillation buffer is turned off and clock input is accepted if the sub clock is not chosen as a CPU clock.
Note 5: When CM05 bit is set to “1, the XOUT pin goes “H”. Furthermore, because the internal feedback resistor remains connected, the XIN pin is pulled “H” to the same level as XOUT via the feedback resistor.
Note 6: After setting the CM04 bit to “1” (XCIN-XCOUT oscillator function), wait until the sub-clock oscillates stably before switching the CM07 bit from “0” to “1” (sub-clock).
Note 7: When entering stop mode from high or middle speed mode, the CM06 bit is set to “1” (divide-by-8 mode).
Note 8: The fC32 clock does not stop. During low speed or low power dissipation mode, do not set this bit to “1” (peripheral clock turned off when in wait mode).
Note 9: To use a sub-clock, set this bit to “1”. Also make sure ports P86 and P87 are directed for input, with no pull-ups.
Note 10: When the PM21 bit of PM2 register is set to “1” (clock modification disable), writing to the CM02, CM05, and CM07 bits has no effect.
Note 11: If the PM21 bit needs to be set to “1”, set the CM07 bit to “0”(main clock) before setting it.
Note 12: To use the main clock as the clock source for the CPU clock, follow the procedure below.
(1) Set the CM05 bit to “0” (oscillate).
(2) Wait until td(M-L) elapses or the main clock oscillation stabilizes, whichever is longer.
(3) Set the CM07 bit all to “0”.
Note 13: When the CM05 bit is set to “1” (main clock turned off), the CM06 bit is fixed to “1” (divide-by-8 mode) and the CM15 bit is fixed to “1” (drive capability high).
Note 14: Keep in mind that the values after reset differ by the input voltage at the STARTB pin.

**Figure 4.2 CM0 Register**

System clock control register 1 (Note 1)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | 0 | 0 | 0 | 0 | |

| Symbol | Address | After reset |
|---|---|---|
| CM1 | $0007_{16}$ | $00100000_2$ |

| Bit symbol | Bit | Function | RW |
|---|---|---|---|
| CM10 | All clock stop control bit (Notes 4, 5) | 0 : Clock on<br>1 : All clocks off (stop mode) | RW |
| —— (b4-b1) | Reserved bit | Must set to “0” | RW |
| CM15 | XIN-XOUT drive capacity select bit (Note 2) | 0 : LOW<br>1 : HIGH | RW |
| CM16 | Main clock division select bit 1 (Note 3) | b7 b6<br>0 0 : No division mode<br>0 1 : Division by 2 mode<br>1 0 : Division by 4 mode<br>1 1 : Division by 16 mode | RW |
| CM17 | | | RW |

Note 1: Write to this register after setting the PRC0 bit of PRCR register to “1” (write enable).
Note 2: When entering stop mode from high or middle speed mode, or when the CM05 bit is set to “1” (main clock turned off) in low speed mode, the CM15 bit is set to “1” (drive capability high).
Note 3: Effective when the CM06 bit is “0” (CM16 and CM17 bits enable).
Note 4: If the CM10 bit is “1” (stop mode), XOUT goes “H” and the internal feedback resistor is disconnected. The XCIN and XCOUT pins are placed in the high-impedance state.
Note 5: When the PM21 bit of PM2 register is set to “1” (clock modification disable), writing to the CM10 bits has no effect.

**Figure 4.3 CM1 Register**

Peripheral clock select register (Note)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | | |

Symbol: PCLKR
Address: 025E16
When reset: 00000011 2

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| PCLK0 | Timers A, B clock select bit (Clock source for the timers A and B | 0 : f2<br>1 : f1 | RW |
| PCLK1 | SI/O clock select bit (Clock source for UART0 to UART2, SI/O3, SI/O4) | 0 : f2SIO<br>1 : f1SIO | RW |
| —— (b7-b2) | Reserved bit | Must set to “0” | RW |

Note: Write to this register after setting the PRC0 bit of PRCR register to “1” (write enable).

Processor mode register 2 (Note 1)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| X | X | X | 0 | 0 | 0 | | 0 |

Symbol: PM2
Address: 001E16
After reset: XXX000002

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| — (b0) | Reserved bit | Must set to “0” | RW |
| PM21 | System clock protective bit (Note 2, Note 3) | 0 : Clock is protected by PRCR register<br>1 : Clock modification disabled | RW |
| — (b4-b2) | Reserved bit | Must set to “0” | RW |
| — (b7-b5) | Nothing is assigned. When write, set to “0”. When read, its content is indeterminate. | | — |

Note 1: Write to this register after setting the PRC1 bit of PRCR register to “1” (write enable).
Note 2: Once this bit is set to “1”, it cannot be cleared to “0” in a program.
Note 3: If the PM21 bit is set to “1,” writing to the following bits has no effect.
CM02 bit of CM0 register
CM05 bit of CM0 register (main clock is not halted)
CM07 bit of CM0 register (CPU clock source does not change)
CM10 bit of CM1 register (stop mode is not entered)

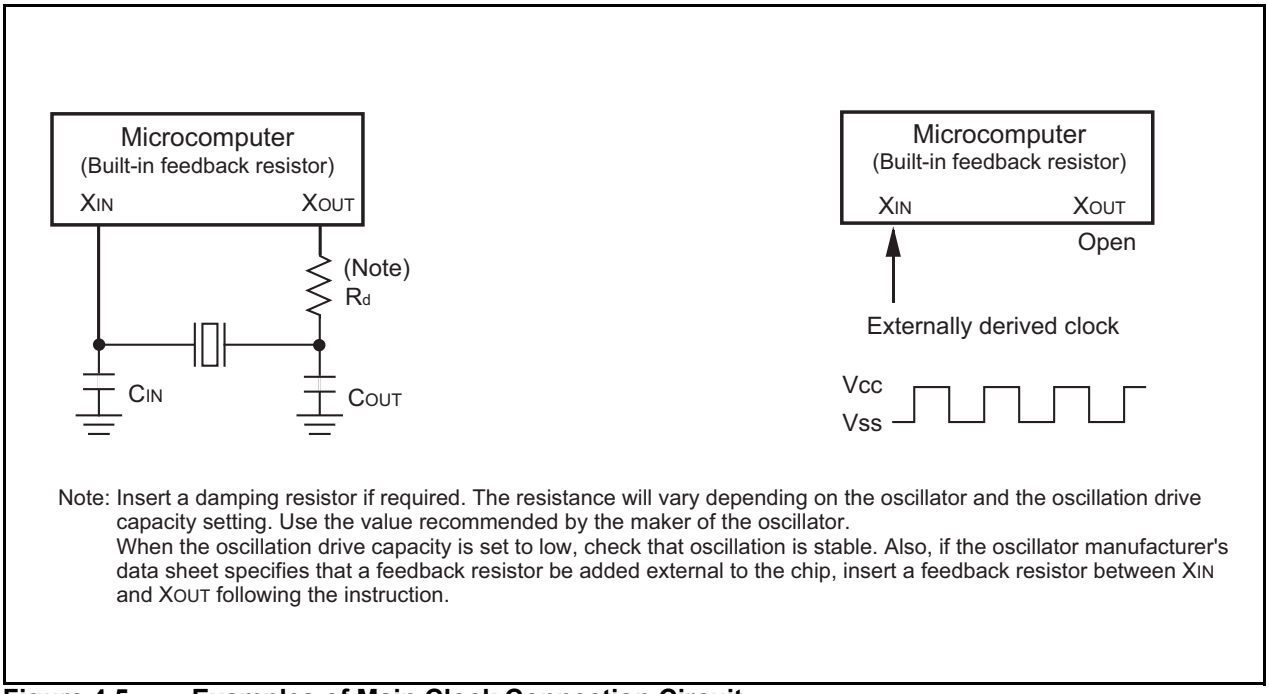
**Figure 4.4 PCLKR Register and PM2 Register**

## 4.1 Oscillator Circuit

The following describes the clocks generated by the clock generation circuit.
Two oscillation circuits are built in the clock generating circuit, and a main clock or a sub clock can be chosen as a CPU clock by setup of the STARTB pin after reset.

### 4.1.1 Main Clock

This clock is used as the clock source for the CPU and peripheral function clocks. The main clock oscillator circuit is configured by connecting a resonator between the XIN and XOUT pins. The main clock oscillator circuit contains a feedback resistor, which is disconnected from the oscillator circuit during stop mode in order to reduce the amount of power consumed in the chip. The main clock oscillator circuit may also be configured by feeding an externally generated clock to the XIN pin. Figure 4.5 shows the examples of main clock connection circuit.
When the level on the STARTB pin is "L", the main clock divided by 8 is selected for the CPU clock (Sub clock turned off) after reset.
The power consumption in the chip can be reduced by setting the CM05 bit of CM0 register to "1" (main clock oscillator circuit turned off) after switching the clock source for the CPU clock to a sub clock. In this case, XOUT goes "H". Furthermore, because the internal feedback resistor remains on, XIN is pulled "H" to XOUT via the feedback resistor. Note that if an externally generated clock is fed into the XIN pin, the main clock cannot be turned off by setting the CM05 bit to "1" without selecting sub clock for the CPU clock. If necessary, use an external circuit to turn off the clock.
During stop mode, all clocks including the main clock are turned off. Refer to "power control".




Note: Insert a damping resistor if required. The resistance will vary depending on the oscillator and the oscillation drive capacity setting. Use the value recommended by the maker of the oscillator.
When the oscillation drive capacity is set to low, check that oscillation is stable. Also, if the oscillator manufacturer's data sheet specifies that a feedback resistor be added external to the chip, insert a feedback resistor between XIN and XOUT following the instruction.

**Figure 4.5 Examples of Main Clock Connection Circuit**

### 4.1.2 Sub Clock

The sub clock is generated by the sub clock oscillation circuit. This clock is used as the clock source for the CPU clock, as well as the timer A and timer B count sources. In addition, an fc clock with the same frequency as that of the sub clock can be output from the CLKOUT pin.

The sub clock oscillator circuit is configured by connecting a crystal resonator between the XCIN and XCOUT pins. The sub clock oscillator circuit contains a feedback resistor, which is disconnected from the oscillator circuit during stop mode in order to reduce the amount of power consumed in the chip. The sub clock oscillator circuit may also be configured by feeding an externally generated clock to the XCIN pin.

Figure 4.6 shows the examples of sub clock connection circuit.

When the level on the STARTB pin is "L ", the sub clock is turned off after reset. At this time, the feedback resistor is disconnected from the oscillator circuit.

To use the sub clock for the CPU clock, set the CM07 bit of CM0 register to "1 " (sub clock) after the sub clock becomes oscillating stably.

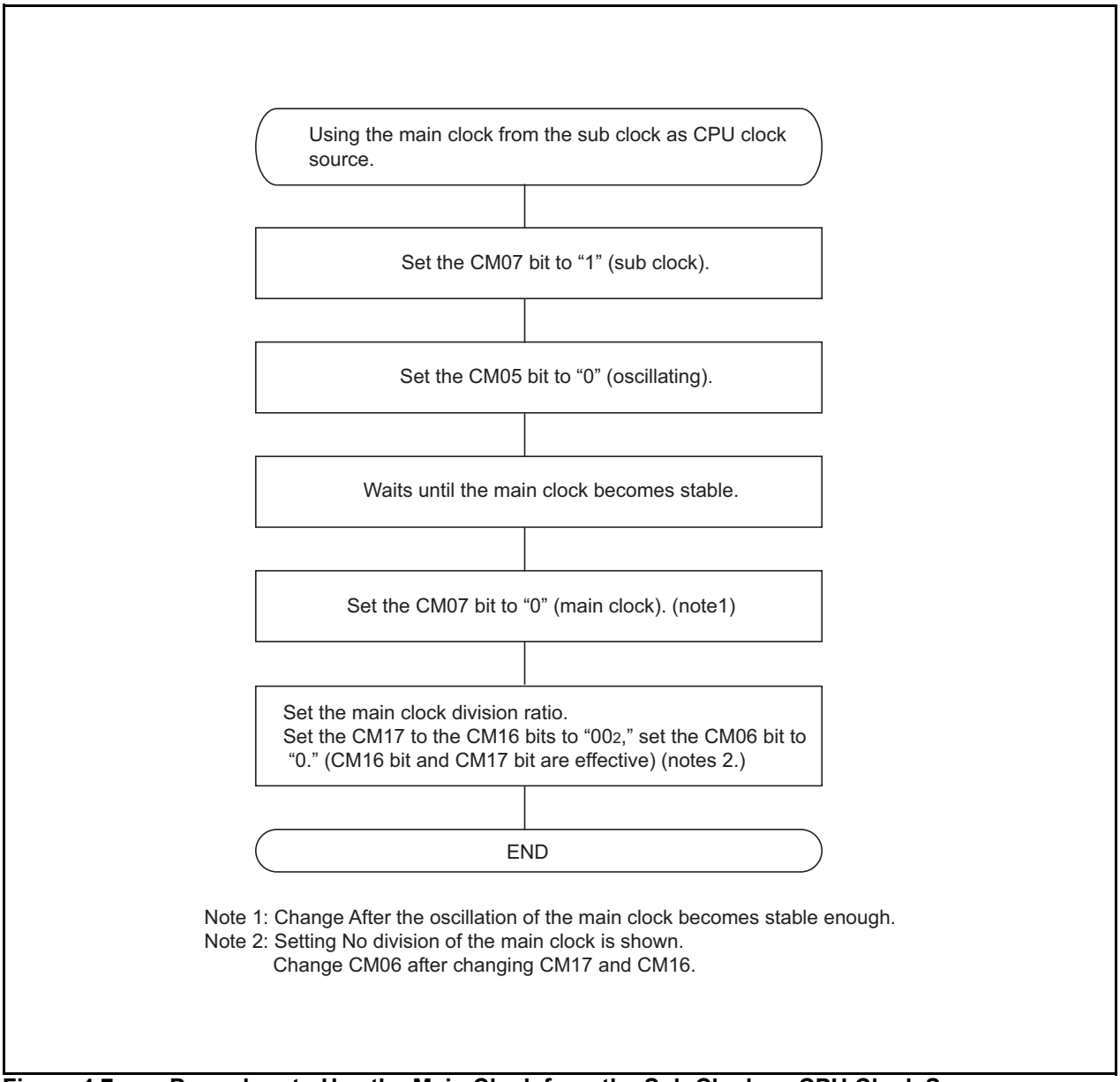
When a STARTB pin is "H ", the sub clock (XCIN) divided by 8 becomes the CPU clock after reset (the main clock stops). When you use a main clock after this, please shift according to the procedure shown in Fig. 4.7.

During stop mode, all clocks including the sub clock are turned off. Refer to "power control".




Note: Insert a damping resistor if required. The resistance will vary depending on the oscillator and the oscillation drive capacity setting. Use the value recommended by the maker of the oscillator.
When the oscillation drive capacity is set to low, check that oscillation is stable. Also, if the oscillator manufacturer's data sheet specifies that a feedback resistor be added external to the chip, insert a feedback resistor between XCIN and XCOUT following the instruction.

**Figure 4.6 Examples of Sub Clock Connection Circuit**




Note 1: Change After the oscillation of the main clock becomes stable enough.
Note 2: Setting No division of the main clock is shown.
Change CM06 after changing CM17 and CM16.

**Figure 4.7 Procedure to Use the Main Clock from the Sub Clock as CPU Clock Source**

## 4.2 CPU Clock and Peripheral Function Clock

Two type clocks: CPU clock to operate the CPU and peripheral function clocks to operate the peripheral functions.

### 4.2.1 CPU Clock and BCLK

These are operating clocks for the CPU and watchdog timer.
The clock source for the CPU clock can be chosen to be the main clock or sub clock.
If the main clock is selected as the clock source for the CPU clock, the selected clock source can be divided by 1 (undivided), 2, 4, 8 or 16 to produce the CPU clock. Use the CM06 bit in CM0 register and the CM17 to CM16 bits in CM1 register to select the divide-by-n value.
When the level on the STARTB pin is "H", the main clock divided by 8 provides the CPU clock after reset.
When the level on the STARTB pin is "L", the sub clock of frequency divided by 8 provides the CPU clock after reset.
At this time, the CM04 bit and the CM05 bit of CM0 register become "1" .
Note that when entering stop mode from high or middle speed mode, or when the CM05 bit of CM0 register is set to "1" (main clock turned off) in low-speed mode, the CM06 bit of CM0 register is set to "1" (divide-by-8 mode).

### 4.2.2 Peripheral Function Clock($f_1$, $f_2$, $f_8$, $f_{32}$, $f_{1SIO}$, $f_{2SIO}$, $f_{8SIO}$, $f_{32SIO}$, $f_{AD}$, $f_{C32}$)

These are operating clocks for the peripheral functions.
Of these, fi (i = 1, 2, 8, 32) and $f_{iSIO}$ are derived from the main clock by dividing them by i. The clock fi is used for timers A and B, and $f_{iSIO}$ is used for serial I/O. The $f_8$ and $f_{32}$ clocks can be output from the $CLK_{OUT}$ pin.
The $f_{AD}$ clock is produced from the main clock, and is used for the A/D converter.
When the WAIT instruction is executed after setting the CM02 bit of CM0 register to "1" (peripheral function clock turned off during wait mode), or when the microcomputer is in low power dissipation mode, the fi, $f_{iSIO}$ and $f_{AD}$ clocks are turned off.
The $f_{C32}$ clock is produced from the sub clock, and is used for timers A and B. This clock can be used when the sub clock is on.

## 4.3 Clock Output Function

During single-chip mode, the $f_8$, $f_{32}$ or $f_C$ clock can be output from the $CLK_{OUT}$ pin. Use the CM01 to CM00 bits of CM0 register to select.

## 4.4 Power Control

There are three power control modes. For convenience' sake, all modes other than wait and stop modes are referred to as normal operation mode here.

### 4.4.1 Normal Operation Mode

Normal operation mode is further classified into four modes.
In normal operation mode, because the CPU clock and the peripheral function clocks both are on, the CPU and the peripheral functions are operating. Power control is exercised by controlling the CPU clock frequency. The higher the CPU clock frequency, the greater the processing capability. The lower the CPU clock frequency, the smaller the power consumption in the chip. If the unnecessary oscillator circuits are turned off, the power consumption is further reduced.
Before the clock sources for the CPU clock can be switched over, the new clock source to which switched must be oscillating stably. If the new clock source is the main clock or sub clock, allow a sufficient wait time in a program until it becomes oscillating stably.

- High-speed Mode
  The main clock divided by 1 provides the CPU clock. If the sub clock is on, $f_{C32}$ can be used as the count source for timers A and B.
- Medium-speed Mode
  The main clock divided by 2, 4, 8 or 16 provides the CPU clock. If the sub clock is on, $f_{C32}$ can be used as the count source for timers A and B.
- Low-speed Mode
  The sub clock provides the CPU clock. The main clock is used as the clock source for the peripheral function clock.
  The $f_{C32}$ clock can be used as the count source for timers A and B.
- Low Power Dissipation Mode
  In this mode, the main clock is turned off after being placed in low speed mode. The sub clock provides the CPU clock. The $f_{C32}$ clock can be used as the count source for timers A and B.
  Simultaneously when this mode is selected, the CM06 bit of CM0 register becomes "1" (divided by 8 mode).
  In the low power dissipation mode, do not change the CM06 bit. Consequently, the medium speed (divided by 8) mode is to be selected when the main clock is operated next.

### 4.4.2 Wait Mode

In wait mode, the CPU clock is turned off, so are the CPU (because operated by the CPU clock) and the watchdog timer. Because the main clock and sub clock are on, the peripheral functions using these clocks keep operating.

- Peripheral Function Clock Stop Function
  If the CM02 bit is "1" (peripheral function clocks turned off during wait mode), the $f_1$, $f_2$, $f_8$, $f_{32}$, $f_{1SIO}$, $f_{8SIO}$, $f_{32SIO}$ and $f_{AD}$ clocks are turned off when in wait mode, with the power consumption reduced that much. However, $f_{C32}$ remains on.
- Entering Wait Mode
  The microcomputer is placed into wait mode by executing the WAIT instruction.
- Pin Status During Wait Mode
  Pin Status During Wait Mode is shown in Table 4.2.
- Exiting Wait Mode
  The microcomputer is moved out of wait mode by a hardware reset, $\overline{\text{NMI}}$ interrupt or peripheral function interrupt.
  If the microcomputer is to be moved out of exit wait mode by a hardware reset or $\overline{\text{NMI}}$ interrupt, set the peripheral function interrupt priority ILVL2 to ILVL0 bits to "$000_2$" (interrupts disabled) before executing the WAIT instruction.
  The peripheral function interrupts are affected by the CM02 bit. If CM02 bit is "0" (peripheral function clocks not turned off during wait mode), all peripheral function interrupts can be used to exit wait mode. If CM02 bit is "1" (peripheral function clocks turned off during wait mode), the peripheral functions using the peripheral function clocks stop operating, so that only the peripheral functions clocked by external signals can be used to exit wait mode.

**Table 4.2 Pin Status During Wait Mode**

| Pin | | Single-chip mode |
|---|---|---|
| $A_0$ to $A_{19}$, $D_0$ to $D_{15}$, $\overline{CS0}$ to $\overline{CS3}$, $\overline{BHE}$ | | |
| $\overline{RD}$, $\overline{WR}$, $\overline{WRL}$, $\overline{WRH}$ | | |
| $\overline{HLDA}$,BCLK | | |
| ALE | | |
| I/O ports | | Retains status before wait mode |
| CLKOUT | When fc selected | Does not stop |
| | When $f_8$, $f_{32}$ selected | Does not stop when the CM02 bit is “0”. When the CM02 bit is “1”, the status immediately prior to entering wait mode is maintained. |

**Table 4.3 Interrupts to Exit Wait Mode**

| Interrupt | CM02=0 | CM02=1 |
|---|---|---|
| $\overline{NMI}$ interrupt | Can be used | Can be used |
| Serial I/O interrupt | Can be used when operating with internal or external clock | Can be used when operating with external clock |
| A-D conversion interrupt | Can be used in one-shot mode or single sweep mode | — (Do not use) |
| Timer A interrupt<br>Timer B interrupt | Can be used in all modes | Can be used in event counter mode or when the count source is fC32 |
| $\overline{INT}$ interrupt | Can be used | Can be used |

Table 4.3 lists the interrupts to exit wait mode.
If the microcomputer is to be moved out of wait mode by a peripheral function interrupt, set up the following before executing the WAIT instruction.

(1) In the ILVL2 to ILVL0 bits of interrupt control register, set the interrupt priority level of the peripheral function interrupt to be used to exit wait mode. Also, for all of the peripheral function interrupts not used to exit wait mode, set the ILVL2 to ILVL0 bits to “$000_2$” (interrupt disable).
(2) Set the I flag to “1”.
(3) Enable the peripheral function whose interrupt is to be used to exit wait mode. In this case, when an interrupt request is generated and the CPU clock is thereby turned on, an interrupt routine is executed.

The CPU clock turned on when exiting wait mode by a peripheral function interrupt is the same CPU clock that was on when the WAIT instruction was executed.

### 4.4.3 Stop Mode

In stop mode, all oscillator circuits are turned off, so are the CPU clock and the peripheral function clocks. Therefore, the CPU and the peripheral functions clocked by these clocks stop operating. The least amount of power is consumed in this mode. If the voltage applied to Vcc pins is VRAM or more, the internal RAM is retained.

However, the peripheral functions clocked by external signals keep operating. The following interrupts can be used to exit stop mode.

- $\overline{\text{NMI}}$ interrupt
- $\overline{\text{INT}}$ interrupt
- Timer A, Timer B interrupt (when counting external pulses in event counter mode)
- Serial I/O interrupt (when external clock is selected)

The internal oscillator circuit of expansion function (Data acquisition / humming function) stops oscillation when expansion register XTAL_VCO, PDC_VCO_ON, VPS_VCO_ON = "L".

- Entering Stop Mode

  The microcomputer is placed into stop mode by setting the CM10 bit of CM1 register to "1" (all clocks turned off). At the same time, the CM06 bit of CM0 register is set to "1" (divide-by-8 mode) and the CM15 bit of CM1 register is set to "1" (main clock oscillator circuit drive capability high).

- Pin Status in Stop Mode

  Table 4.4 lists pin status during stop mode

- Exiting Stop Mode

The microcomputer is moved out of stop mode by a hardware reset, $\overline{\text{NMI}}$ interrupt or peripheral function interrupt.

If the microcomputer is to be moved out of stop mode by a hardware reset or $\overline{\text{NMI}}$ interrupt, set the peripheral function interrupt priority ILVL2 to ILVL0 bits to "000$_2$" (interrupts disable) before setting the CM10 bit to "1".

If the microcomputer is to be moved out of stop mode by a peripheral function interrupt, set up the following before setting the CM10 bit to "1".

(1) In the ILVL2 to ILVL0 bits of interrupt control register, set the interrupt priority level of the peripheral function interrupt to be used to exit stop mode.
Also, for all of the peripheral function interrupts not used to exit stop mode, set the ILVL2 to ILVL0 bits to "000$_2$".

(2) Set the I flag to "1".

(3) Enable the peripheral function whose interrupt is to be used to exit stop mode.
In this case, when an interrupt request is generated and the CPU clock is thereby turned on, an interrupt service routine is executed.

Which CPU clock will be used after exiting stop mode by a peripheral function or NMI interrupt is determined by the CPU clock that was on when the microcomputer was placed into stop mode as follows:

If the CPU clock before entering stop mode was derived from the sub clock: sub clock

If the CPU clock before entering stop mode was derived from the main clock: main clock divide-by-8

**Table 4.4 Pin Status in Stop Mode**

| Pin | | Single-chip mode |
|---|---|---|
| $A_0$ to $A_{19}$, $D_0$ to $D_{15}$, $\overline{CS0}$ to $\overline{CS3}$, $\overline{BHE}$ | | |
| $\overline{RD}$, $\overline{WR}$, $\overline{WRL}$, $\overline{WRH}$ | | |
| $\overline{HLDA}$, BCLK | | |
| ALE | | |
| I/O ports | | Retains status before stop mode |
| $CLK_{OUT}$ | When fc selected | "H" |
| | When $f_8$, $f_{32}$ selected | Retains status before stop mode |

Figure 4.8 shows the state transition from normal operation mode to stop mode and wait mode.
Figure 4.9 shows the state transition in normal operation mode.

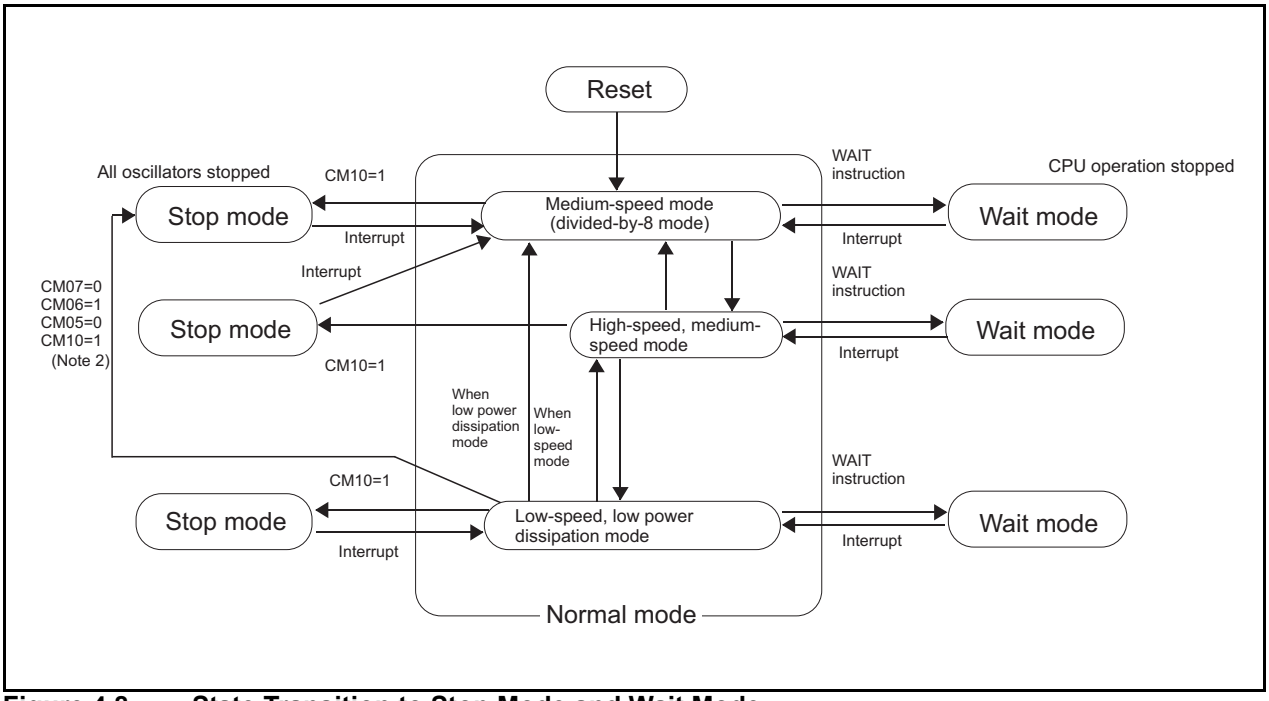
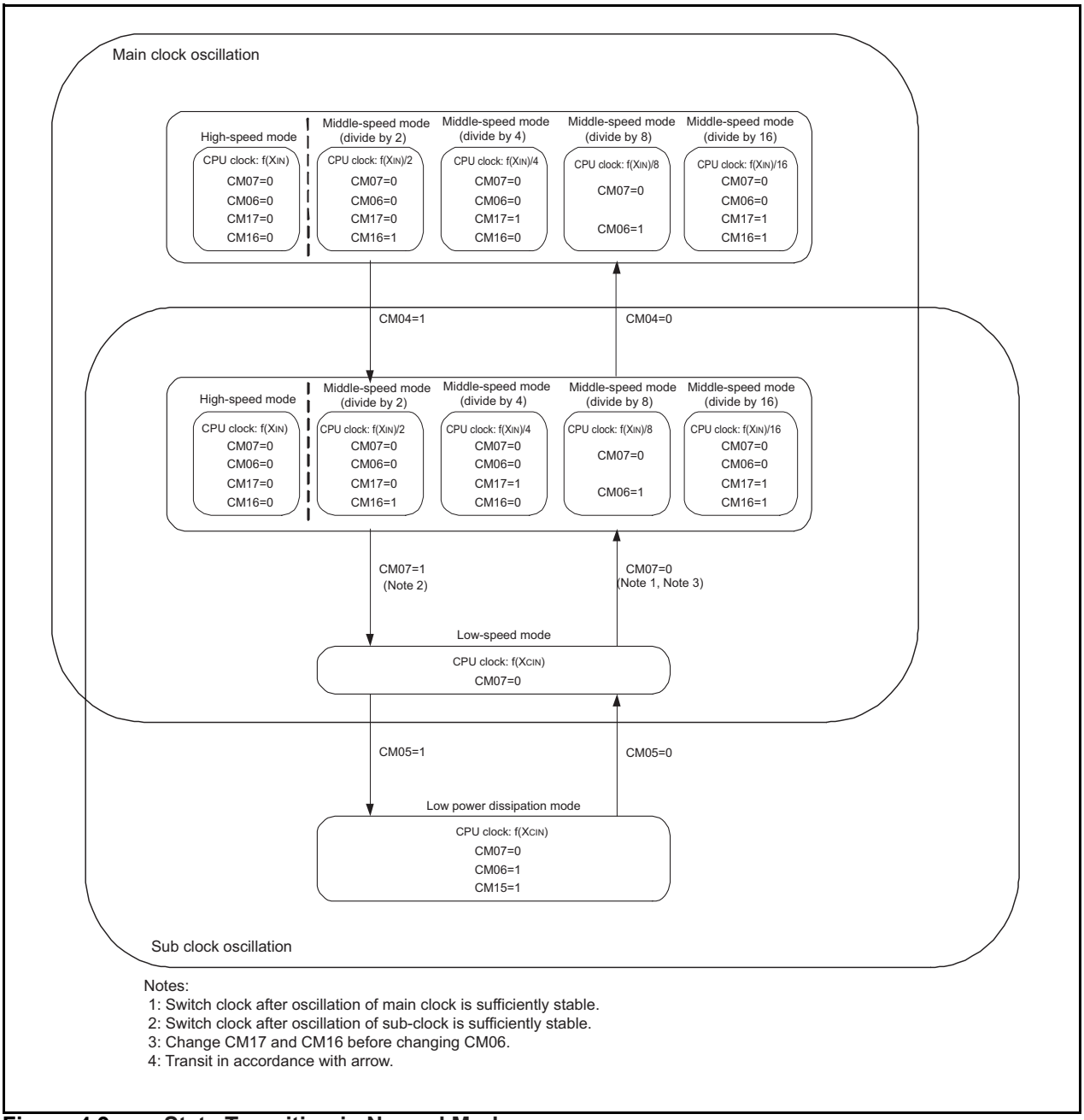



**Figure 4.8 State Transition to Stop Mode and Wait Mode**




Notes:
1: Switch clock after oscillation of main clock is sufficiently stable.
2: Switch clock after oscillation of sub-clock is sufficiently stable.
3: Change CM17 and CM16 before changing CM06.
4: Transit in accordance with arrow.

**Figure 4.9 State Transition in Normal Mode**

## 4.5 System Clock Protective Function

When the main clock is selected for the CPU clock source, this function disables the clock against modifications in order to prevent the CPU clock from becoming halted by run-away.
If the PM21 bit of PM2 register is set to "1" (clock modification disabled), the following bits are protected against writes:

- CM02, CM05, and CM07 bits in CM0 register
- CM10, CM11 bits in CM1 register

Before the system clock protective function can be used, the following register settings must be made while the CM05 bit of CM0 register is "0" (main clock oscillating) and CM07 bit is "0" (main clock selected for the CPU clock source):

(1) Set the PRC1 bit of PRCR register to "1" (enable writes to PM2 register).
(2) Set the PM21 bit of PM2 register to "1" (disable clock modification).
(3) Set the PRC1 bit of PRCR register to "0" (disable writes to PM2 register).

Do not execute the WAIT instruction when the PM21 bit is "1".

# 5. Protection

In the event that a program runs out of control, this function protects the important registers so that they will not be rewritten easily. Figure 5.1 shows the PRCR register. The following lists the registers protected by the PRCR register.

- Registers protected by PRC0 bit: CM0, CM1 and PCLKR registers
- Registers protected by PRC1 bit: PM0, PM1 and PM2 registers
- Registers protected by PRC2 bit: PD9, S3C and S4C registers

Set the PRC2 bit to "1" (write enabled) and then write to any address, and the PRC2 bit will be cleared to "0" (write protected). The registers protected by the PRC2 bit should be changed in the next instruction after setting the PRC2 bit to "1". Make sure no interrupts or DMA transfers will occur between the instruction in which the PRC2 bit is set to "1" and the next instruction. The PRC0 and PRC1 bits are not automatically cleared to "0" by writing to any address. They can only be cleared in a program.

Protect register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| X | X | 0 | 0 | 0 | | | |

Symbol: PRCR
Address: $000A_{16}$
After reset: $XX000000_2$

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| PRC0 | Protect bit 0 | Enable write to CM0, CM1 and PCLKR registers<br>0 : Write protected<br>1 : Write enabled | RW |
| PRC1 | Protect bit 1 | Enable write to PM0, PM1 and PM2 registers<br>0 : Write protected<br>1 : Write enabled | RW |
| PRC2 | Protect bit 2 | Enable write to PD9, S3C and S4C registers<br>0 : Write protected<br>1 : Write enabled | RW |
| ——— (b5-b3) | Reserved bit | Must set to "0" | RW |
| ——— (b7-b6) | Nothing is assigned. When write, set to "0". When read, its content is interdeterminate. | | — |

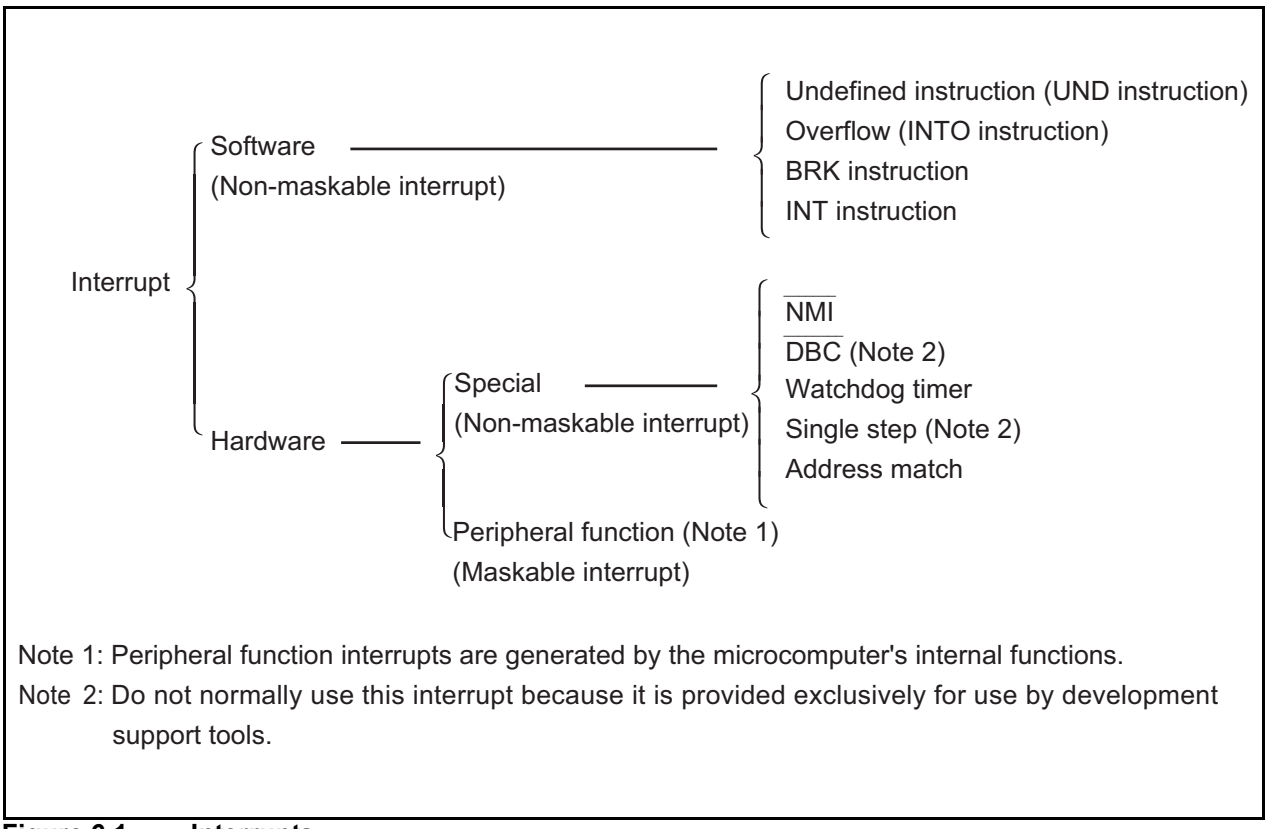
Note: The PRC2 bit is set to "0" by writing to any address after setting it to "1". Other bits are not set to "0" by writing to any address, and must therefore be set in a program.

**Figure 5.1 PRCR Register**

# 6. Interrupts

## 6.1 Type of Interrupts

Figure 6.1 shows types of interrupts.




Note 1: Peripheral function interrupts are generated by the microcomputer's internal functions.
Note 2: Do not normally use this interrupt because it is provided exclusively for use by development support tools.

**Figure 6.1 Interrupts**

- Maskable Interrupt: An interrupt which can be enabled (disabled) by the interrupt enable flag (I flag) or whose interrupt priority can be changed by priority level.
- Non-maskable Interrupt: An interrupt which cannot be enabled (disabled) by the interrupt enable flag (I flag) or whose interrupt priority cannot be changed by priority level.

## 6.2 Software Interrupts

A software interrupt occurs when executing certain instructions. Software interrupts are non-maskable interrupts.

- Undefined Instruction Interrupt
  An undefined instruction interrupt occurs when executing the UND instruction.
- Overflow Interrupt
  An overflow interrupt occurs when executing the INTO instruction with the O flag set to "1" (the operation resulted in an overflow). The following are instructions whose O flag changes by arithmetic:
  ABS, ADC, ADCF, ADD, CMP, DIV, DIVU, DIVX, NEG, RMPA, SBB, SHA, SUB
- BRK Interrupt
  A BRK interrupt occurs when executing the BRK instruction.
- INT Instruction Interrupt
  An INT instruction interrupt occurs when executing the INT instruction. Software interrupt Nos. 0 to 63 can be specified for the INT instruction. Because software interrupt Nos. 4 to 31 are assigned to peripheral function interrupts, the same interrupt routine as for peripheral function interrupts can be executed by executing the INT instruction.
  In software interrupt Nos. 0 to 31, the U flag is saved to the stack during instruction execution and is cleared to "0" (ISP selected) before executing an interrupt sequence. The U flag is restored from the stack when returning from the interrupt routine. In software interrupt Nos. 32 to 63, the U flag does not change state during instruction execution, and the SP then selected is used.

## 6.3 Hardware Interrupts

Hardware interrupts are classified into two types — special interrupts and peripheral function interrupts.

• Special Interrupts

Special interrupts are non-maskable interrupts.

(1) $\overline{\text{NMI}}$ Interrupt

An $\overline{\text{NMI}}$ interrupt is generated when input on the $\overline{\text{NMI}}$ pin changes state from high to low. For details about the $\overline{\text{NMI}}$ interrupt, refer to the section "NMI interrupt".

(2) $\overline{\text{DBC}}$ Interrupt

Do not normally use this interrupt because it is provided exclusively for use by development support tools.

(3) Watchdog Timer Interrupt

Generated by the watchdog timer. Once a watchdog timer interrupt is generated, be sure to initialize the watchdog timer. For details about the watchdog timer, refer to the section "watchdog timer".

(4) Single-step Interrupt

Do not normally use this interrupt because it is provided exclusively for use by development support tools.

(5) Address Match Interrupt

An address match interrupt is generated immediately before executing the instruction at the address indicated by the RMAD0 to RMAD3 register that corresponds to one of the AIER register's AIER0 or AIER1 bit or the AIER2 register's AIER20 or AIER21 bit which is "1" (address match interrupt enabled). For details about the address match interrupt, refer to the section "address match interrupt".

• Peripheral Function Interrupts

Peripheral function interrupts are maskable interrupts and generated by the microcomputer's internal functions. The interrupt sources for peripheral function interrupts are listed in Table 6.2. For details about the peripheral functions, refer to the description of each peripheral function in this manual.

## 6.4 Interrupts and Interrupt Vector

One interrupt vector consists of 4 bytes. Set the start address of each interrupt routine in the respective interrupt vectors. When an interrupt request is accepted, the CPU branches to the address set in the corresponding interrupt vector. Figure 6.2 shows the interrupt vector.




**Figure 6.2 Interrupt Vector**

• Fixed Vector Tables

The fixed vector tables are allocated to the addresses from FFFDC16 to FFFFF16. Table 6.1 lists the fixed vector tables. In the flash memory version of microcomputer, the vector addresses (H) of fixed vectors are used by the ID code check function. For details, refer to the section "flash memory rewrite disabling function".

**Table 6.1 Fixed Vector Tables**

| Interrupt source | Vector table addresses Address (L) to address (H) | Remarks | Reference |
|---|---|---|---|
| Undefined instruction | FFFDC16 to FFFDF16 | Interrupt on UND instruction | M16C/60, M16C/20 series software manual |
| Overflow | FFFE016 to FFFE316 | Interrupt on INTO instruction | |
| BRK instruction | FFFE416 to FFFE716 | If the contents of address FFFE716 is FF16, program execution starts from the address shown by the vector in the relocatable vector table. | |
| Address match | FFFE816 to FFFEB16 | | Address match interrupt |
| Single step (Note) | FFFEC16 to FFFEF16 | | |
| Watchdog timer | FFFF016 to FFFF316 | | Watchdog timer |
| $\overline{\text{DBC}}$ (Note) | FFFF416 to FFFF716 | | |
| NMI | FFFF816 to FFFFB16 | | $\overline{\text{NMI}}$ interrupt |
| Reset | FFFFC16 to FFFFF16 | | Reset |

Note: Do not normally use this interrupt because it is provided exclusively for use by development support tools.

• Relocatable Vector Tables

The 256 bytes beginning with the start address set in the INTB register comprise a reloacatable vector table area. Table 6.2 lists the relocatable vector tables. Setting an even address in the INTB register results in the interrupt sequence being executed faster than in the case of odd addresses.

**Table 6.2 Relocatable Vector Tables**

| Interrupt source | Vector address (Note 1) Address (L) to address (H) | Software interrupt number | Reference |
|---|---|---|---|
| BRK instruction (Note 5) | +0 to +3 ($0000_{16}$ to $0003_{16}$) | 0 | M16C/60, M16C/20 series software manual |
| ——— (Reserved) | | 1 to 3 | |
| $\overline{INT3}$ | +16 to +19 ($0010_{16}$ to $0013_{16}$) | 4 | $\overline{INT}$ interrupt |
| Timer B5/SLICE ON (Note 7) | +20 to +23 ($0014_{16}$ to $0017_{16}$) | 5 | Timer |
| Timer B4/Remote control, UART1 bus collision detect (Note 4, Note 6, Note 7) | +24 to +27 ($0018_{16}$ to $001B_{16}$) | 6 | Timer Serial I/O |
| Timer B3/HINT, UART0 bus collision detect (Note 4, Note 6, Note 7) | +28 to +31 ($001C_{16}$ to $001F_{16}$) | 7 | |
| SI/O4, $\overline{INT5}$ (Note 2) | +32 to +35 ($0020_{16}$ to $0023_{16}$) | 8 | $\overline{INT}$ interrupt Serial I/O |
| SI/O3, $\overline{INT4}$ (Note 2) | +36 to +39 ($0024_{16}$ to $0027_{16}$) | 9 | |
| UART 2 bus collision detection | +40 to +43 ($0028_{16}$ to $002B_{16}$) | 10 | Serial I/O |
| DMA0 | +44 to +47 ($002C_{16}$ to $002F_{16}$) | 11 | DMAC |
| DMA1 | +48 to +51 ($0030_{16}$ to $0033_{16}$) | 12 | |
| A/D | +56 to +59 ($0038_{16}$ to $003B_{16}$) | 14 | A/D converter |
| UART2 transmit, NACK2 (Note 3) | +60 to +63 ($003C_{16}$ to $003F_{16}$) | 15 | Serial I/O |
| UART2 receive, ACK2 (Note 3) | +64 to +67 ($0040_{16}$ to $0043_{16}$) | 16 | |
| UART0 transmit, NACK0 (Note 3) | +68 to +71 ($0044_{16}$ to $0047_{16}$) | 17 | |
| UART0 receive, ACK0 (Note 3) | +72 to +75 ($0048_{16}$ to $004B_{16}$) | 18 | |
| UART1 transmit, NACK1 (Note 3) | +76 to +79 ($004C_{16}$ to $004F_{16}$) | 19 | |
| UART1 receive, ACK1 (Note 3) | +80 to +83 ($0050_{16}$ to $0053_{16}$) | 20 | |
| Timer A0 | +84 to +87 ($0054_{16}$ to $0057_{16}$) | 21 | Timer |
| Timer A1 | +88 to +91 ($0058_{16}$ to $005B_{16}$) | 22 | |
| Timer A2 | +92 to +95 ($005C_{16}$ to $005F_{16}$) | 23 | |
| Timer A3 | +96 to +99 ($0060_{16}$ to $0063_{16}$) | 24 | |
| Timer A4/Multi-master $I^2C$ (Note 9) | +100 to +103 ($0064_{16}$ to $0067_{16}$) | 25 | |
| Timer B0 | +104 to +107 ($0068_{16}$ to $006B_{16}$) | 26 | |
| Timer B1 | +108 to +111 ($006C_{16}$ to $006F_{16}$) | 27 | |
| Timer B2/Clock timer (Note 7) | +112 to +115 ($0070_{16}$ to $0073_{16}$) | 28 | |
| $\overline{INT0}$ | +116 to +119 ($0074_{16}$ to $0077_{16}$) | 29 | $\overline{INT}$ interrupt |
| $\overline{INT1}$ | +120 to +123 ($0078_{16}$ to $007B_{16}$) | 30 | |
| $\overline{INT2}$/Remote control transmission (Note 8) | +124 to +127 ($007C_{16}$ to $007F_{16}$) | 31 | |
| Software interrupt (Note 5) | +128 to +131 ($0080_{16}$ to $0083_{16}$) to +252 to +255 ($00FC_{16}$ to $00FF_{16}$) | 32 to 63 | M16C/60, M16C/20 series software manual |

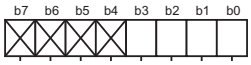
Notes 1: Address relative to address in INTB
Notes 2: Use the IFSR register's IFSR6 and IFSR7 bits to select.
Notes 3: During $I^2C$ mode, NACK and ACK interrupts comprise the interrupt source.
Notes 4: Use the IFSR2A register's IFSR26 and IFSR27 bits to select.
Notes 5: These interrupts cannot be disabled using the I flag.
Notes 6: Bus collision detection : During IE mode, this bus collision detection constitutes the cause of an interrupt. During $I^2C$ mode, however, a start condition or a stop condition detection constitutes the cause of an interrupt.
Notes 7: When you use SLICEON, remote control, HINT and clock timer interruption, refer to address $36_{16}$ expansion register of "14. Expansion Function"
Notes 8: Please refer to address $3E_{16}$ of the expansion register of "14. Expansion Function" when you use the remote control transmission interrupt.
Notes 9: Please refer to the $I^2C0$ interrupt control register of "11 multi-master $I^2C$-BUS interface" (address $02D6_{16}$) when you use multi master $I^2C$ interrupt.

## 6.5 Interrupt Control

The following describes how to enable/disable the maskable interrupts, and how to set the priority in which order they are accepted. What is explained here does not apply to nonmaskable interrupts.
Use the FLG register's I flag, IPL, and each interrupt control register's ILVL2 to ILVL0 bits to enable/disable the maskable interrupts. Whether an interrupt is requested is indicated by the IR bit in each interrupt control register.
Figure 6.3 shows the interrupt control registers.

Interrupt control register (Note 2)

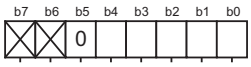
| Symbol | Address | After reset |
|---|---|---|
| TB5IC | $0045_{16}$ | $XXXXX000_2$ |
| TB4IC/U1BCNIC (Note 3) | $0046_{16}$ | $XXXXX000_2$ |
| TB3IC/U0BCNIC (Note 3) | $0047_{16}$ | $XXXXX000_2$ |
| BCNIC | $004A_{16}$ | $XXXXX000_2$ |
| DM0IC, DM1IC | $004B_{16}$, $004C_{16}$ | $XXXXX000_2$ |
| ADIC | $004E_{16}$ | $XXXXX000_2$ |
| S0TIC to S2TIC | $0051_{16}$, $0053_{16}$, $004F_{16}$ | $XXXXX000_2$ |
| S0RIC to S2RIC | $0052_{16}$, $0054_{16}$, $0050_{16}$ | $XXXXX000_2$ |
| TA0IC to TA4IC | $0055_{16}$ to $0059_{16}$ | $XXXXX000_2$ |
| TB0IC to TB2IC | $005A_{16}$ to $005C_{16}$ | $XXXXX000_2$ |




| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| ILVL0 | Interrupt priority level select bit | b2 b1 b0<br>0 0 0 : Level 0 (interrupt disabled)<br>0 0 1 : Level 1<br>0 1 0 : Level 2<br>0 1 1 : Level 3<br>1 0 0 : Level 4<br>1 0 1 : Level 5<br>1 1 0 : Level 6<br>1 1 1 : Level 7 | RW |
| ILVL1 | | | RW |
| ILVL2 | | | RW |
| IR | Interrupt request bit | 0 : Interrupt not requested<br>1 : Interrupt requested | RW<br>(Note 1) |
| —<br>(b7-b4) | No functions are assigned.<br>When writing to these bits, write “0”. The values in these bits when read are indeterminate. | | — |

Note 1: This bit can only be reset by writing "0" (Do not write "1").
Note 2: To rewrite the interrupt control registers, do so at a point that does not generate the interrupt request for that register. For details, see the precautions for interrupts.
Note 3: Use the IFSR2A register to select.

| Symbol | Address | After reset |
|---|---|---|
| INT3IC | $0044_{16}$ | $XX00X000_2$ |
| S4IC/INT5IC | $0048_{16}$ | $XX00X000_2$ |
| S3IC/INT4IC | $0049_{16}$ | $XX00X000_2$ |
| INT0IC to INT2IC | $005D_{16}$ to $005F_{16}$ | $XX00X000_2$ |




| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| ILVL0 | Interrupt priority level select bit | b2 b1 b0<br>0 0 0 : Level 0 (interrupt disabled)<br>0 0 1 : Level 1<br>0 1 0 : Level 2<br>0 1 1 : Level 3<br>1 0 0 : Level 4<br>1 0 1 : Level 5<br>1 1 0 : Level 6<br>1 1 1 : Level 7 | RW |
| ILVL1 | | | RW |
| ILVL2 | | | RW |
| IR | Interrupt request bit | 0: Interrupt not requested<br>1: Interrupt requested | RW<br>(Note 1) |
| POL | Polarity select bit | 0 : Selects falling edge (Notes 3, 4)<br>1 : Selects rising edge | RW |
| — | Reserved bit | Must always be set to “0” | RW |
| —<br>(b7-b6) | No functions are assigned.<br>When writing to these bits, write “0”. The values in these bits when read are indeterminate. | | RW |

Note 1: This bit can only be reset by writing "0" (Do not write "1").
Note 2: To rewrite the interrupt control register, do so at a point that does not generate the interrupt request for that register. For details, see the precautions for interrupts.
Note 3: If the IFSR register’s IFSRi bit (i = 0 to 5) is "1" (both edges), set the INTiIC register’s POL bit to "0 "(falling edge).
Note 4: Set the S3IC or S4IC register’s POL bit to "0" (falling edge) when the IFSR register’s IFSR6 bit = 0 (SI/O3 selected) or IFSR7 bit = 0 (SI/O4 selected), respectively

**Figure 6.3 Interrupt Control Registers**

## 6.6 I Flag

The I flag enables or disables the maskable interrupt. Setting the I flag to "1" (= enabled) enables the maskable interrupt. Setting the I flag to "0" (= disabled) disables all maskable interrupts.

## 6.7 IR Bit

The IR bit is set to "1" (= interrupt requested) when an interrupt request is generated. Then, when the interrupt request is accepted and the CPU branches to the corresponding interrupt vector, the IR bit is cleared to "0" (= interrupt not requested).
The IR bit can be cleared to "0" in a program. Note that do not write "1" to this bit.

## 6.8 ILVL2 to ILVL0 Bits and IPL

Interrupt priority levels can be set using the ILVL2 to ILVL0 bits.
Table 6.3 shows the settings of interrupt priority levels and Table 6.4 shows the interrupt priority levels enabled by the IPL.

The following are conditions under which an interrupt is accepted:

- I flag = "1"
- IR bit = "1"
- interrupt priority level > IPL

The I flag, IR bit, ILVL2 to ILVL0 bits and IPL are independent of each other. In no case do they affect one another.

**Table 6.3 Settings of Interrupt Priority Levels**

| ILVL2 to ILVL0 bits | Interrupt priority level | Priority order |
|---|---|---|
| $000_2$ | Level 0 (interrupt disabled) | ——— |
| $001_2$ | Level 1 | Low ↓ |
| $010_2$ | Level 2 | |
| $011_2$ | Level 3 | |
| $100_2$ | Level 4 | |
| $101_2$ | Level 5 | |
| $110_2$ | Level 6 | |
| $111_2$ | Level 7 | High |

**Table 6.4 Interrupt Priority Levels Enabled by IPL**

| IPL | Enabled interrupt priority levels |
|---|---|
| $000_2$ | Interrupt levels 1 and above are enabled |
| $001_2$ | Interrupt levels 2 and above are enabled |
| $010_2$ | Interrupt levels 3 and above are enabled |
| $011_2$ | Interrupt levels 4 and above are enabled |
| $100_2$ | Interrupt levels 5 and above are enabled |
| $101_2$ | Interrupt levels 6 and above are enabled |
| $110_2$ | Interrupt levels 7 and above are enabled |
| $111_2$ | All maskable interrupts are disabled |

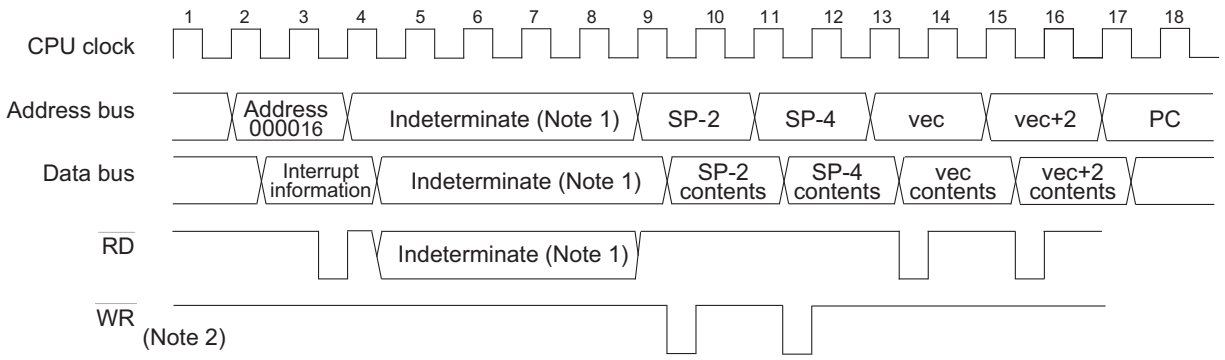
## 6.9 Interrupt Sequence

An interrupt sequence — what are performed over a period from the instant an interrupt is accepted to the instant the interrupt routine is executed — is described here.
If an interrupt occurs during execution of an instruction, the processor determines its priority when the execution of the instruction is completed, and transfers control to the interrupt sequence from the next cycle. If an interrupt occurs during execution of either the SMOVB, SMOVF, SSTR or RMPA instruction, the processor temporarily suspends the instruction being executed, and transfers control to the interrupt sequence.
The CPU behavior during the interrupt sequence is described below. Figure 6.4 shows time required for executing the interrupt sequence.

(1) The CPU gets interrupt information (interrupt number and interrupt request priority level) by reading the address 0000016. Then it clears the IR bit for the corresponding interrupt to "0" (interrupt not requested).
(2) The FLG register immediately before entering the interrupt sequence is saved to the CPU's internal temporary register(Note 1).
(3) The I, D and U flags in the FLG register become as follows:
The I flag is cleared to "0" (interrupts disabled).
The D flag is cleared to "0" (single-step interrupt disabled).
The U flag is cleared to "0" (ISP selected).
However, the U flag does not change state if an INT instruction for software interrupt Nos. 32 to 63 is executed.
(4) The CPU's internal temporary register (Note 1) is saved to the stack.
(5) The PC is saved to the stack.
(6) The interrupt priority level of the accepted interrupt is set in the IPL.
(7) The start address of the relevant interrupt routine set in the interrupt vector is stored in the PC.

After the interrupt sequence is completed, the processor resumes executing instructions from the start address of the interrupt routine.

Note: This register cannot be used by user.

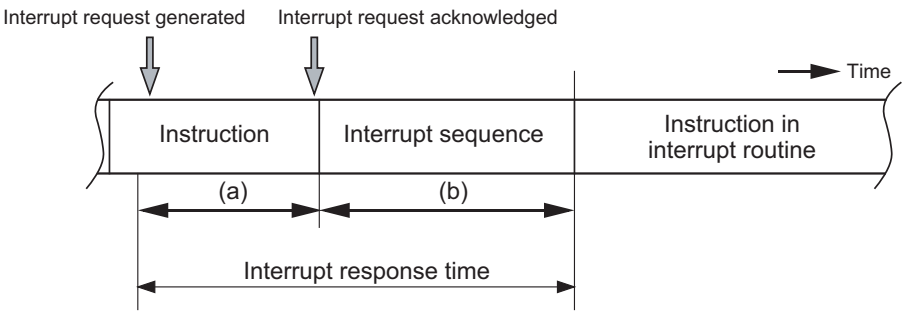



Note 1 : The indeterminate state depends on the instruction queue buffer. A read cycle occurs when the instruction queue buffer is ready to accept instructions.
Note 2 : The WR signal timing shown here is for the case where the stack is located in the internal RAM.

**Figure 6.4 Time Required for Executing Interrupt Sequence**

## 6.10 Interrupt Response Time

Figure 6.5 shows the interrupt response time. The interrupt response or interrupt acknowledge time denotes a time from when an interrupt request is generated till when the first instruction in the interrupt routine is executed. Specifically, it consists of a time from when an interrupt request is generated till when the instruction then executing is completed ((a) in Figure 6.5) and a time during which the interrupt sequence is executed ((b) in Figure 6.5).




(a) A time from when an interrupt request is generated till when the instruction then executing is completed. The length of this time varies with the instruction being executed. The DIVX instruction requires the longest time, which is equal to 30 cycles (without wait state, the divisor being a register).

(b) A time during which the interrupt sequence is executed. For details, see the table below. Note, however, that the values in this table must be increased 2 cycles for the $\overline{\text{DBC}}$ interrupt and 1 cycle for the address match and single-step interrupts.

| Interrupt vector address | SP value | 16-Bit bus, without wait | 8-Bit bus, without wait |
|---|---|---|---|
| Even | Even | 18 cycles | 20 cycles |
| Even | Odd | 19 cycles | 20 cycles |
| Odd | Even | 19 cycles | 20 cycles |
| Odd | Odd | 20 cycles | 20 cycles |

**Figure 6.5 Interrupt response time**

## 6.11 Variation of IPL when Interrupt Request is Accepted

When a maskable interrupt request is accepted, the interrupt priority level of the accepted interrupt is set in the IPL. When a software interrupt or special interrupt request is accepted, one of the interrupt priority levels listed in Table 6.5 is set in the IPL. Shown in Table 6.5 are the IPL values of software and special interrupts when they are accepted.

**Table 6.5 IPL Level That is Set to IPL When A Software or Special Interrupt Is Accepted**

| Interrupt sources | Level that is set to IPL |
|---|---|
| Watchdog timer, $\overline{\text{NMI}}$ | 7 |
| Software, address match, $\overline{\text{DBC}}$, single-step | Not changed |

## 6.12 Saving Registers

In the interrupt sequence, the FLG register and PC are saved to the stack.
At this time, the 4 high-order bits of the PC and the 4 high-order (IPL) and 8 low-order bits of the FLG register, 16 bits in total, are saved to the stack first. Next, the 16 low-order bits of the PC are saved. Figure 6.6 shows the stack status before and after an interrupt request is accepted.
The other necessary registers must be saved in a program at the beginning of the interrupt routine. Use the PUSHM instruction, and all registers except SP can be saved with a single instruction.

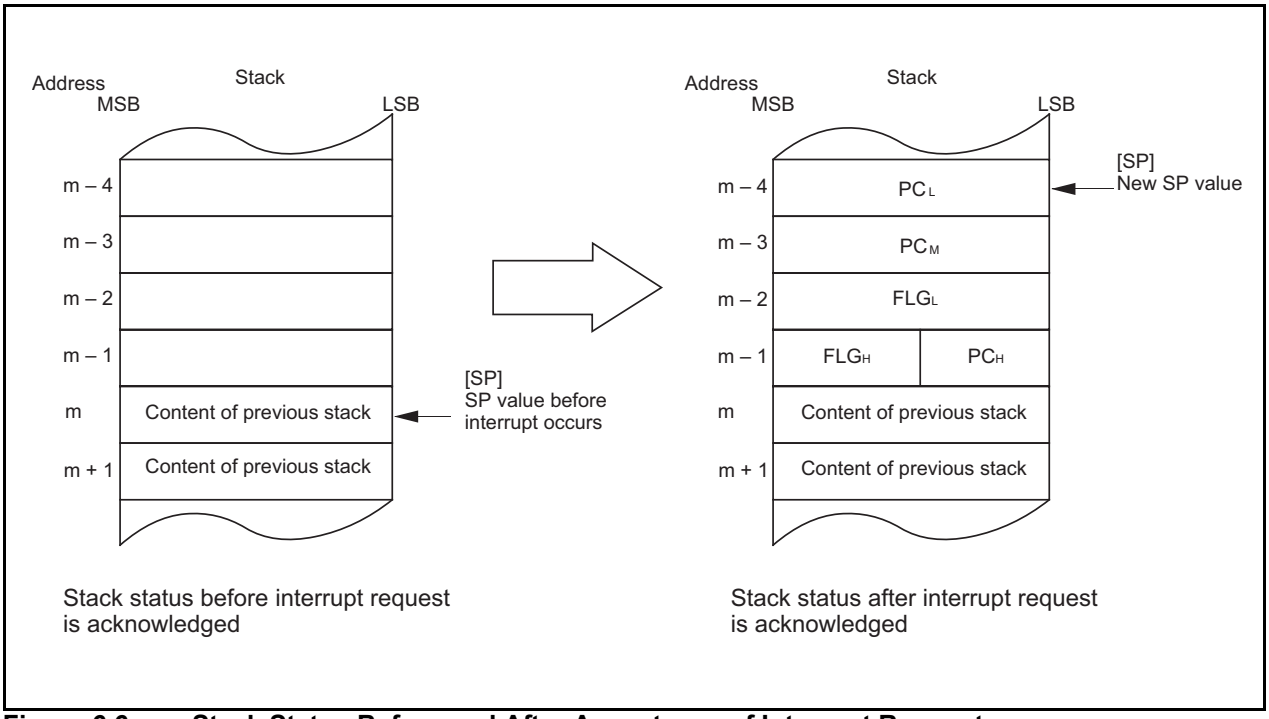
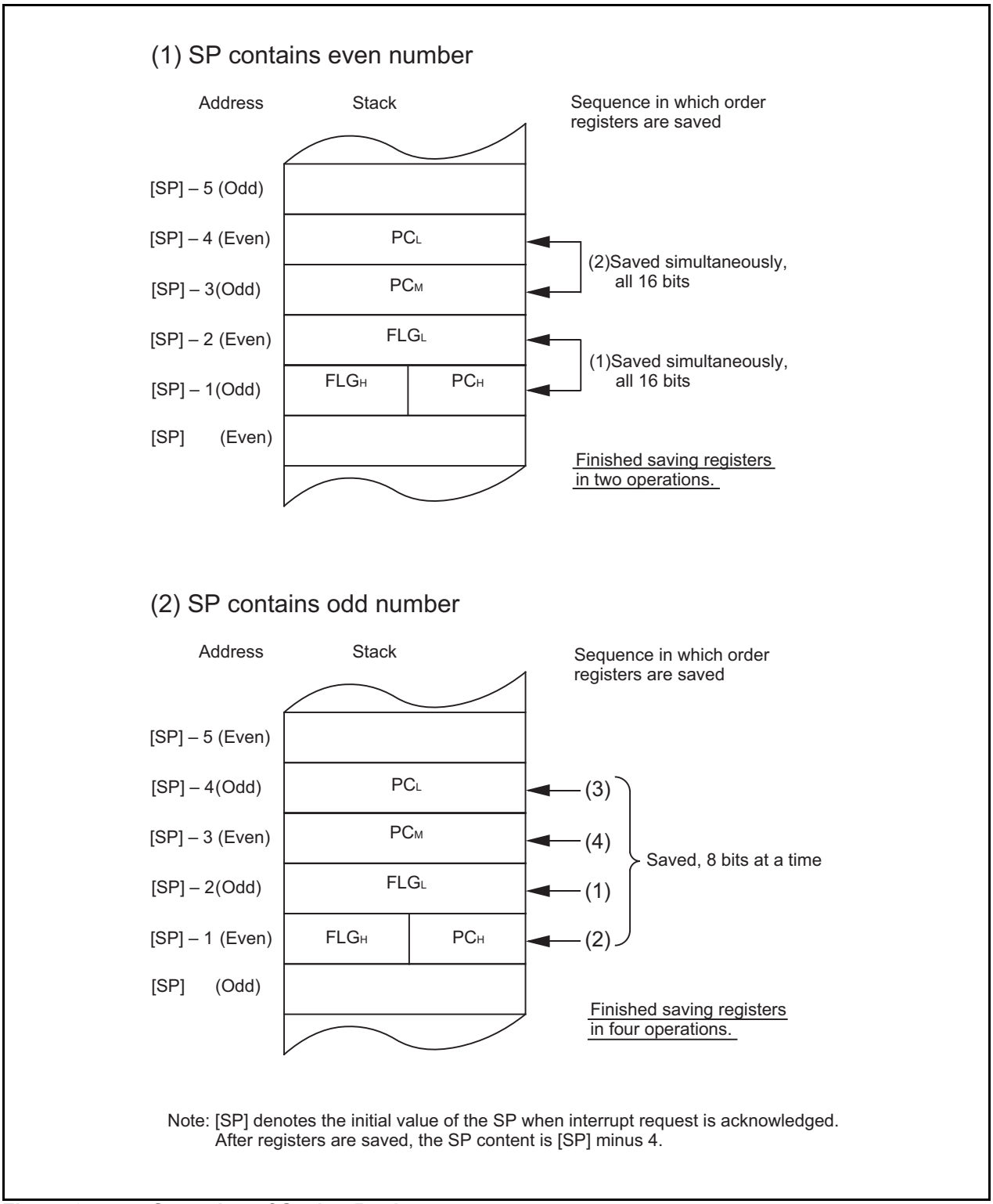



**Figure 6.6 Stack Status Before and After Acceptance of Interrupt Request**

The operation of saving registers carried out in the interrupt sequence is dependent on whether the SP(Note), at the time of acceptance of an interrupt request, is even or odd. If the stack pointer (Note) is even, the FLG register and the PC are saved, 16 bits at a time. If odd, they are saved in two steps, 8 bits at a time. Figure 6.7 shows the operation of the saving registers.

Note: When any INT instruction in software numbers 32 to 63 has been executed, this is the SP indicated by the U flag. Otherwise, it is the ISP.




Note: [SP] denotes the initial value of the SP when interrupt request is acknowledged. After registers are saved, the SP content is [SP] minus 4.

**Figure 6.7 Operation of Saving Register**

## 6.13 Returning from an Interrupt Routine

The FLG register and PC in the state in which they were immediately before entering the interrupt sequence are restored from the stack by executing the REIT instruction at the end of the interrupt routine.
Thereafter the CPU returns to the program which was being executed before accepting the interrupt request.
Return the other registers saved by a program within the interrupt routine using the POPM or similar instruction before executing the REIT instruction

## 6.14 Interrupt Priority

If two or more interrupt requests are generated while executing one instruction, the interrupt request that has the highest priority is accepted.
For maskable interrupts (peripheral functions), any desired priority level can be selected using the ILVL2 to ILVL0 bits. However, if two or more maskable interrupts have the same priority level, their interrupt priority is resolved by hardware, with the highest priority interrupt accepted.
The watchdog timer and other special interrupts have their priority levels set in hardware. Figure 6.8 shows the priorities of hardware interrupts.
Software interrupts are not affected by the interrupt priority. If an instruction is executed, control branches invariably to the interrupt routine.

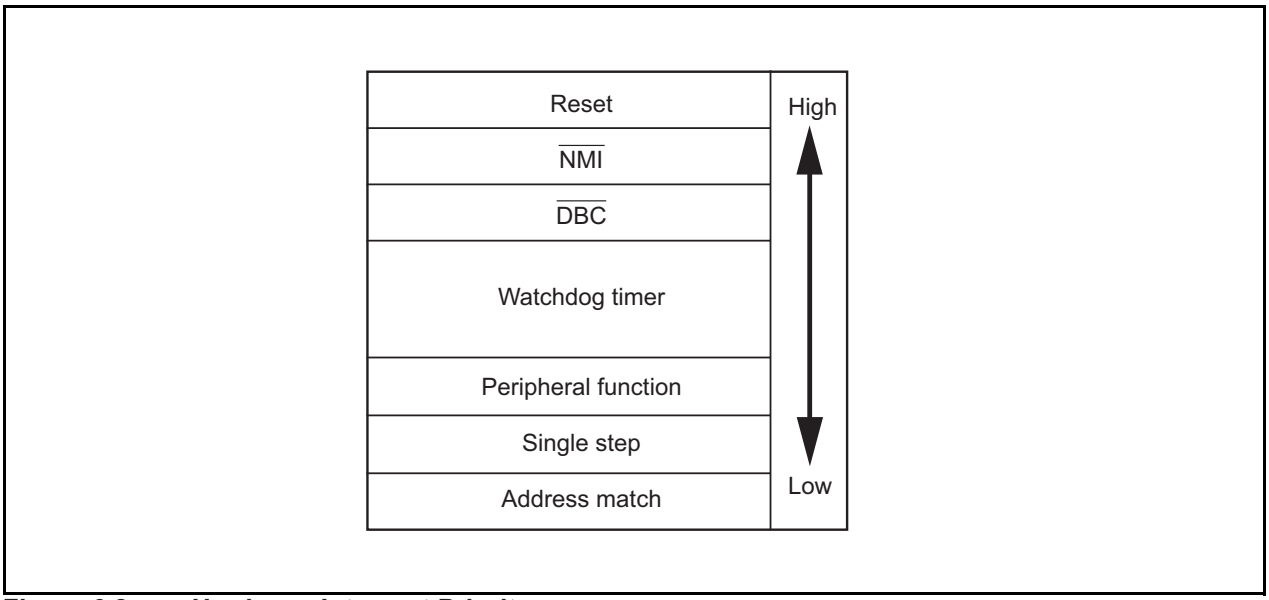



**Figure 6.8 Hardware Interrupt Priority**

## 6.15 Interrupt Priority Resolution Circuit

The interrupt priority resolution circuit is used to select the interrupt with the highest priority among those requested.
Figure 6.9 shows the circuit that judges the interrupt priority level.

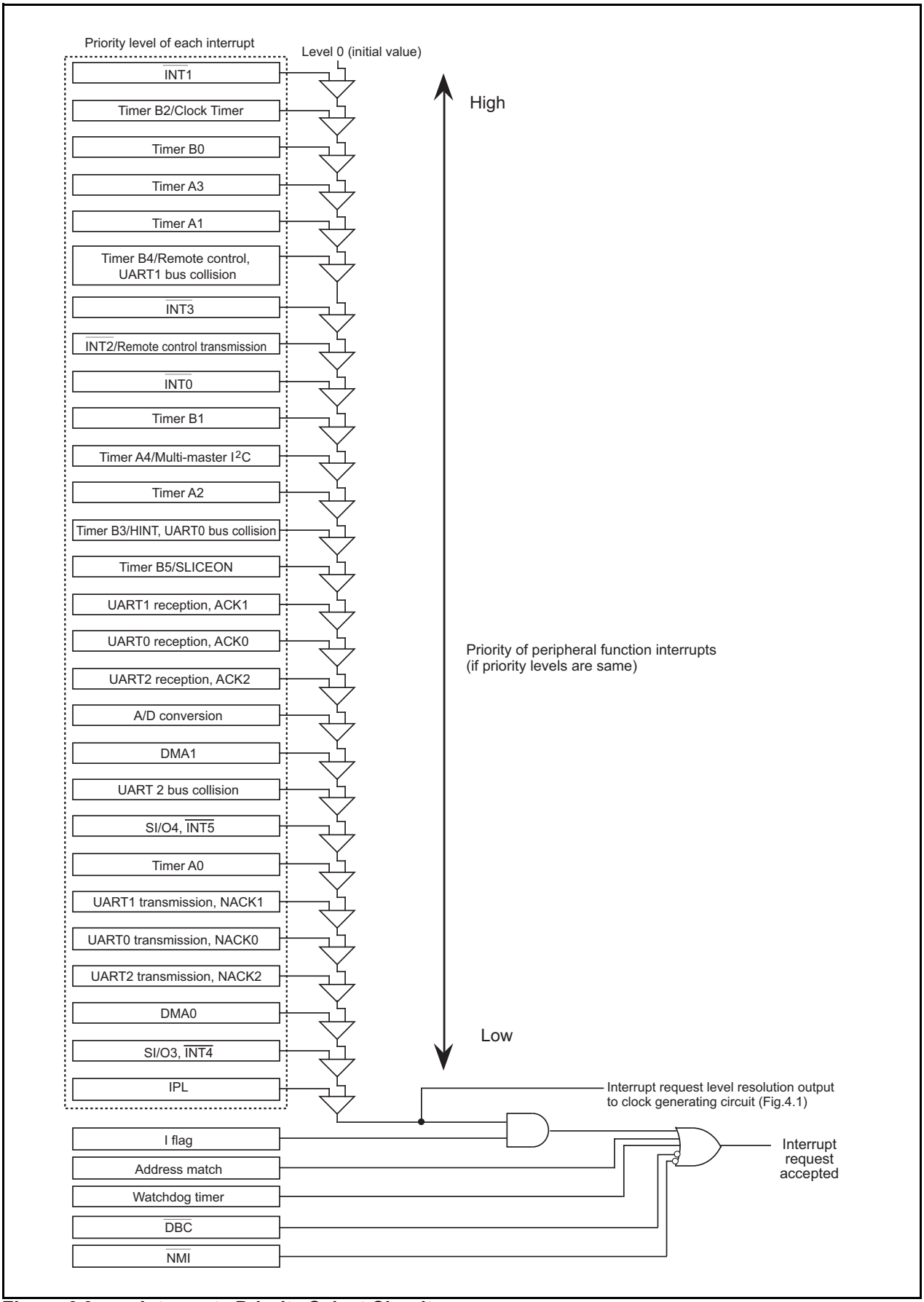



**Figure 6.9 Interrupts Priority Select Circuit**

## 6.16 $\overline{\text{INT}}$ Interrupt

$\overline{\text{INT}}$i interrupt (i = 0 to 5) is triggered by the edges of external inputs. The edge polarity is selected using the IFSR register's IFSRi bit.

$\overline{\text{INT4}}$ and $\overline{\text{INT5}}$ share the interrupt vector and interrupt control register with SI/O3 and SI/O4, respectively.

To use the $\overline{\text{INT4}}$ interrupt, set the IFSR register's IFSR6 bit to "1" (= $\overline{\text{INT4}}$). To use the $\overline{\text{INT5}}$ interrupt, set the IFSR register's IFSR7 bit to "1" (= $\overline{\text{INT5}}$).

After modifying the IFSR6 or IFSR7 bit, clear the corresponding IR bit to "0" (= interrupt not requested) before enabling the interrupt.

$\overline{\text{INT2}}$ and the remote control transmission, the vector and the interrupt control register are shared. (Please refer to "14. Expansion Function" for details. )

Figure 6.10 shows the IFSR and IFSR2A registers.

Interrupt request cause select register

b7 b6 b5 b4 b3 b2 b1 b0

Symbol: IFSR | Address: 035F16 | After reset: 0016

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| IFSR0 | INT0 interrupt polarity switching bit | 0 : One edge<br>1 : Both edges (Note 1) | RW |
| IFSR1 | INT1 interrupt polarity switching bit | 0 : One edge<br>1 : Both edges (Note 1) | RW |
| IFSR2 | INT2 interrupt polarity switching bit | 0 : One edge<br>1 : Both edges (Note 1) | RW |
| IFSR3 | INT3 interrupt polarity switching bit | 0 : One edge<br>1 : Both edges (Note 1) | RW |
| IFSR4 | INT4 interrupt polarity switching bit | 0 : One edge<br>1 : Both edges (Note 1) | RW |
| IFSR5 | INT5 interrupt polarity switching bit | 0 : One edge<br>1 : Both edges (Note 1) | RW |
| IFSR6 | Interrupt request cause select bit | 0 : SI/O3 (Note 2)<br>1 : $\overline{\text{INT4}}$ | RW |
| IFSR7 | Interrupt request cause select bit | 0 : SI/O4 (Note 2)<br>1 : $\overline{\text{INT5}}$ | RW |

Note 1: When setting this bit to "1" (= both edges), make sure the INT0IC to INT5IC register's POL bit is set to "0" (= falling edge).

Note 2: When setting this bit to "0" (= SI/O3, SI/O4), make sure the S3IC and S4IC registers' POL bit is set to "0" (= falling edge).

Interrupt request cause select register 2

b7 b6 b5 b4 b3 b2 b1 b0

Symbol: IFSR2A | Address: 035E16 | After reset: 00XXXXXX2

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| — (b5-b0) | Nothing is assigned. When write, set to "0". When read, their contents are indeterminate. | | — |
| IFSR26 | Interrupt request cause select bit (Note 1) | 0 : Timer B3/HINT<br>1 : UART0 bus collision detection | RW |
| IFSR27 | Interrupt request cause select bit (Note 2) | 0 : Timer B4/Remote control<br>1 : UART1 bus collision detection | RW |

Note 1: Timer B3/HINT and UART0 bus collision detection share the vector and interrupt control register. When using the timer B3/HINT interrupt, clear the IFSR26 bit to "0" (timer B3/HINT). When using UART0 bus collision detection, set the IFSR26 bit to "1".

Note 2: Timer B4/Remote control and UART1 bus collision detection share the vector and interrupt control register. When using the timer B4/Remote control interrupt, clear the IFSR27 bit to "0" (timer B4/Remote control). When using UART1 bus collision detection, set the IFSR27 bit to "1".

**Figure 6.10 IFSR Register and IFSR2A Register**

## 6.17 $\overline{\text{NMI}}$ Interrupt

An $\overline{\text{NMI}}$ interrupt is generated when input on the $\overline{\text{NMI}}$ pin changes state from high to low. The $\overline{\text{NMI}}$ interrupt is a non-maskable interrupt.
The input level of this $\overline{\text{NMI}}$ interrupt input pin can be read by accessing the P8 register's P8_5 bit.
This pin cannot be used as an input port.

## 6.18 Address Match Interrupt

An address match interrupt request is generated immediately before executing the instruction at the address indicated by the RMADi register (i = 0 to 3). Set the start address of any instruction in the RMADi register. Use the AIER register's AIER0 and AIER1 bits and the AIER2 register's AIER20 and AIER21 bits to enable or disable the interrupt. Note that the address match interrupt is unaffected by the I flag and IPL.
For address match interrupts, the value of the PC that is saved to the stack area varies depending on the instruction being executed (refer to "Saving Registers").
(The value of the PC that is saved to the stack area is not the correct return address.) Therefore, follow one of the methods described below to return from the address match interrupt.
• Rewrite the content of the stack and then use the REIT instruction to return.
• Restore the stack to its previous state before the interrupt request was accepted by using the POP or similar other instruction and then use a jump instruction to return.
Table 6.6 shows the value of the PC that is saved to the stack area when an address match interrupt request is accepted.
Note that when using the external bus in 8 bits width, no address match interrupts can be used for externa areas.
Figure 6.11 shows the AIER, AIER2, and RMAD0 to RMAD3 registers.

**Table 6.6 Instruction Just Before Execution and Address Stored in Stack When There Occurs Interrupts**

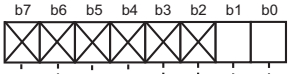
| Instruction at the address indicated by the RMADi register | Value of the PC that is saved to the stack area |
|---|---|
| • 16-bit op-code instruction<br>• Instruction shown below among 8-bit operation code instructions<br>ADD.B:S #IMM8,dest SUB.B:S #IMM8,dest AND.B:S #IMM8,dest<br>OR.B:S #IMM8,dest MOV.B:S #IMM8,dest STZ.B:S #IMM8,dest<br>STNZ.B:S #IMM8,dest STZX.B:S #IMM81,#IMM82,dest<br>CMP.B:S #IMM8,dest PUSHM src POPM dest<br>JMPS #IMM8 JSRS #IMM8<br>MOV.B:S #IMM,dest (However, dest=A0 or A1) | The address indicated by the RMADi register +2 |
| Instructions other than the above | The address indicated by the RMADi register +1 |

Value of the PC that is saved to the stack area : Refer to "Saving Registers".

**Table 6.7 Relationship Between Address Match Interrupt Sources and Associated Registers**

| Address match interrupt sources | Address match interrupt enable bit | Address match interrupt register |
|---|---|---|
| Address match interrupt 0 | AIER0 | RMAD0 |
| Address match interrupt 1 | AIER1 | RMAD1 |
| Address match interrupt 2 | AIER20 | RMAD2 |
| Address match interrupt 3 | AIER21 | RMAD3 |

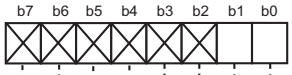
Address match interrupt enable register




| Symbol | Address | After reset |
|---|---|---|
| AIER | $0009_{16}$ | $XXXXXX00_2$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| AIER0 | Address match interrupt 0 enable bit | 0 : Interrupt disabled<br>1 : Interrupt enabled | |
| AIER1 | Address match interrupt 1 enable bit | 0 : Interrupt disabled<br>1 : Interrupt enabled | RW |
| ——<br>(b7-b2) | Nothing is assigned.<br>When write, set to “0”.<br>When read, their contents are indeterminate. | | — |

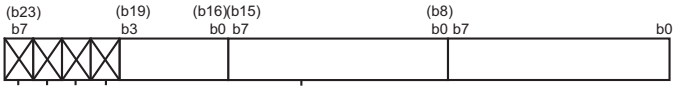
Address match interrupt enable register 2




| Symbol | Address | After reset |
|---|---|---|
| AIER2 | $01BB_{16}$ | $XXXXXX00_2$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| AIER20 | Address match interrupt 2 enable bit | 0 : Interrupt disable<br>1 : Interrupt enabled | RW |
| AIER21 | Address match interrupt 3 enable bit | 0 : Interrupt disabled<br>1 : Interrupt enable | RW |
| ——<br>(b7-b2) | Nothing is assigned.<br>When write, set to “0”.<br>When read, their contents are indeterminate. | | — |

Address match interrupt register i (i = 0 to 3)




| Symbol | Address | After reset |
|---|---|---|
| RMAD0 | $0012_{16}$ to $0010_{16}$ | $X0000000_{16}$ |
| RMAD1 | $0016_{16}$ to $0014_{16}$ | $X0000000_{16}$ |
| RMAD2 | $01BA_{16}$ to $01B8_{16}$ | $X0000000_{16}$ |
| RMAD3 | $01BE_{16}$ to $01BC_{16}$ | $X0000000_{16}$ |

| Function | Setting range | RW |
|---|---|---|
| Address setting register for address match interrupt | $00000_{16}$ to $FFFFF_{16}$ | RW |
| Nothing is assigned.<br>When write, set to “0”.<br>When read, their contents are indeterminate. | | — |

**Figure 6.11 AIER Register, AIER2 Register and RMAD0 to RMAD3 Registers**

# 7. Watchdog Timer

The watchdog timer is the function of detecting when the program is out of control. Therefore, we recommend using the watchdog timer to improve reliability of a system. The watchdog timer contains a 15-bit counter which counts down the clock derived by dividing the CPU clock using the prescaler. Whether to generate a watchdog timer interrupt request or apply a watchdog timer reset as an operation to be performed when the watchdog timer underflows after reaching the terminal count can be selected using the PM12 bit of PM1 register. The PM12 bit can only be set to "1" (reset). Once this bit is set to "1", it cannot be set to "0" (watchdog timer interrupt) in a program.
Refer to "Watchdog Timer Reset" for the details of watchdog timer reset.
When the main clock is selected for CPU clock, the divide-by-N value for the prescaler can be chosen to be 16 or 128 using the WDC7 bit of WDC register. If a sub-clock is selected for CPU clock, the divide-by- N value for the prescaler is always 2 no matter how the WDC7 bit is set. The period of watchdog timer can be calculated as given below. The period of watchdog timer is, however, subject to an error due to the prescaler.

With main clock chosen for CPU clock

$$\text{Watchdog timer period} = \frac{\text{Prescaler dividing (16 or 128) X Watchdog timer count (32768)}}{\text{CPU clock}}$$

With sub-clock chosen for CPU clock

$$\text{Watchdog timer period} = \frac{\text{Prescaler dividing (2) X Watchdog timer count (32768)}}{\text{CPU clock}}$$

For example, when CPU clock = 10 MHz and the divide-by-N value for the prescaler= 16, the watchdog timer period is approx. 52.4 ms.

The watchdog timer is initialized by writing to the WDTS register. The prescaler is initialized after reset.
Note that the watchdog timer and the prescaler both are inactive after reset, so that the watchdog timer is activated to start counting by writing to the WDTS register.
In stop mode, wait mode and hold state, the watchdog timer and prescaler are stopped. Counting is resumed from the held value when the modes or state are released.
Figure 7.1 shows the block diagram of the watchdog timer. Figure 7.2 shows the watchdog timer-related registers.

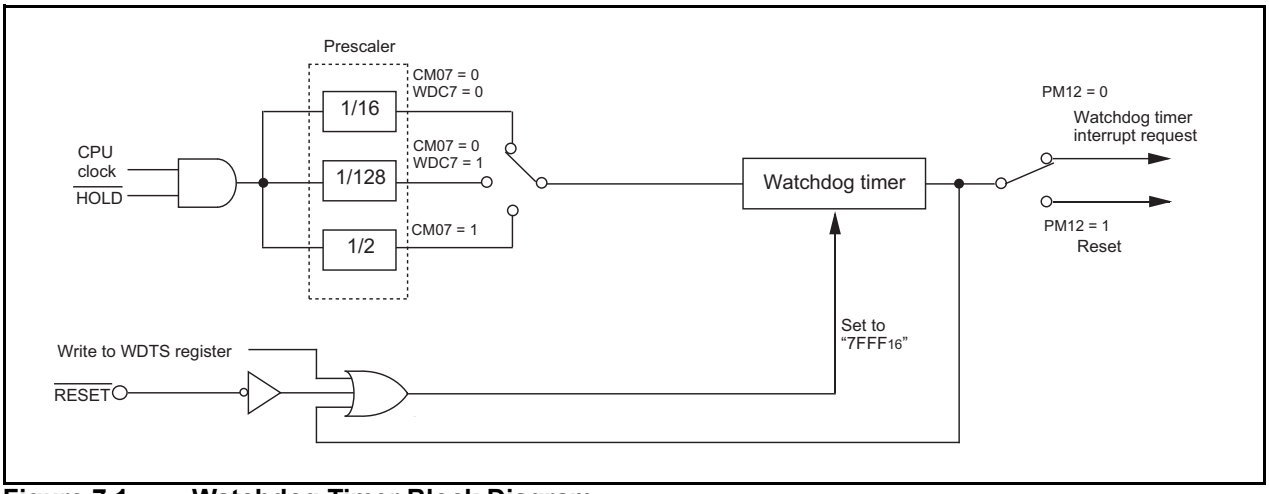



**Figure 7.1 Watchdog Timer Block Diagram**

Watchdog timer control register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | 0 | | | | | | |

Symbol: WDC | Address: 000F16 | After reset: 00XXXXXX2(Note2)

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| $\overline{(b4\text{-}b0)}$ | High-order bit of watchdog timer | | RO |
| WDC5 | Cold start / warm start discrimination flag (Note 1) | 0 : Cold start<br>1 : Warm start | RW |
| $\overline{(b6)}$ | Reserved bit | Must set to "0" | RW |
| WDC7 | Prescaler select bit | 0 : Divided by 16<br>1 : Divided by 128 | RW |

Note 1: The WDC5 bit is always "1" (warm start) no matter how it is set by writing a "0" or "1".
Note 2: The WDC5 bit is "0" (cold start) immediately after power-on. It can only be set to "1" in a program.

Watchdog timer start register (Note)

| b7 | b0 |
|---|---|
| | |

Symbol: WDTS | Address: 000E16 | After reset: Indeterminate

| Function | RW |
|---|---|
| The watchdog timer is initialized and starts counting after a write instruction to this register. The watchdog timer value is always initialized to "7FFF16" regardless of whatever value is written. | WO |

Note : Write to the WDTS register after the watchdog timer interrupt occurs.

**Figure 7.2 WDC Register and WDTS Register**

# 8. DMAC

The DMAC (Direct Memory Access Controller) allows data to be transferred without the CPU intervention.
Two DMAC channels are included. Each time a DMA request occurs, the DMAC transfers one (8 or 16-bit) data from the source address to the destination address. The DMAC uses the same data bus as used by the CPU. Because the DMAC has higher priority of bus control than the CPU and because it makes use of a cycle steal method, it can transfer one word (16 bits) or one byte (8 bits) of data within a very short time after a DMA request is generated. Figure 8.1 shows the block diagram of the DMAC.
Table 8.1 shows the DMAC specifications. Figures 8.2 to 8.4 show the DMAC-related registers.

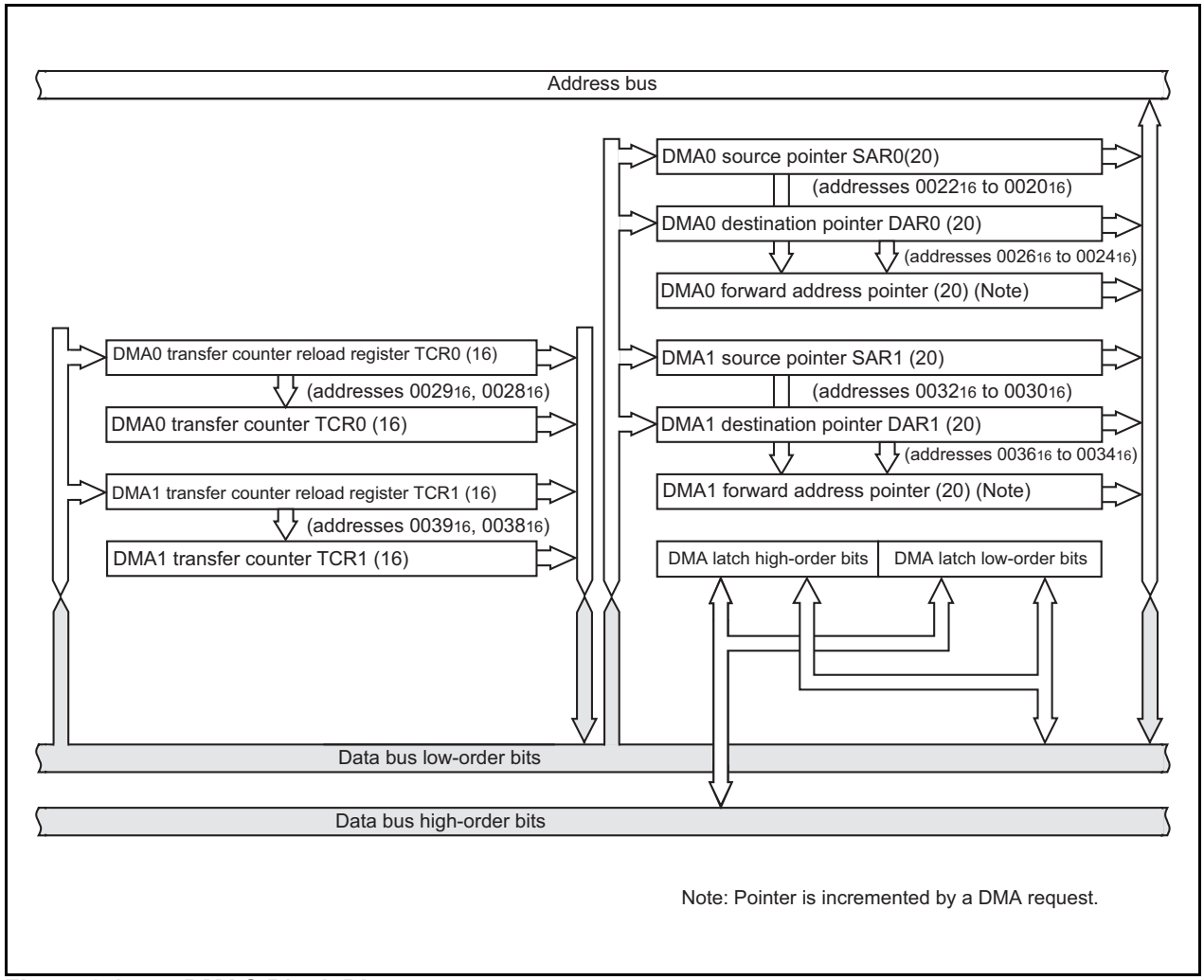



**Figure 8.1 DMAC Block Diagram**

A DMA request is generated by a write to the DMiSL register (i = 0 to 1)'s DSR bit, as well as by an interrupt request which is generated by any function specified by the DMiSL register's DMS and DSEL3 to DSEL0 bits. However, unlike in the case of interrupt requests, DMA requests are not affected by the I flag and the interrupt control register, so that even when interrupt requests are disabled and no interrupt request can be accepted, DMA requests are always accepted. Furthermore, because the DMAC does not affect interrupts, the interrupt control register's IR bit does not change state due to a DMA transfer.
A data transfer is initiated each time a DMA request is generated when the DMiCON register's DMAE bit = "1" (DMA enabled). However, if the cycle in which a DMA request is generated is faster than the DMA transfer cycle, the number of transfer requests generated and the number of times data is transferred may not match. For details, refer to "DMA Requests".

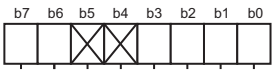
**Table 8.1 DMAC Specifications**

| Item | | Specification |
|---|---|---|
| No. of channels | | 2 (cycle steal method) |
| Transfer memory space | | • From any address in the 1M bytes space to a fixed address<br>• From a fixed address to any address in the 1M bytes space<br>• From a fixed address to a fixed address |
| Maximum No. of bytes transferred | | 128K bytes (with 16-bit transfers) or 64K bytes (with 8-bit transfers) |
| DMA request factors (Note 1, Note 2) | | Falling edge of $\overline{INT0}$ or $\overline{INT1}$<br>Both edge of $\overline{INT0}$ or $\overline{INT1}$<br>Timer A0 to timer A4 interrupt requests<br>Timer B0 to timer B5 interrupt requests<br>UART0 transfer, UART0 reception interrupt requests<br>UART1 transfer, UART1 reception interrupt requests<br>UART2 transfer, UART2 reception interrupt requests<br>SI/O3, SI/O4 interrpt requests<br>A/D conversion interrupt requests<br>Software triggers |
| Channel priority | | DMA0 > DMA1 (DMA0 takes precedence) |
| Transfer unit | | 8 bits or 16 bits |
| Transfer address direction | | forward or fixed (The source and destination addresses cannot both be in the forward direction.) |
| Transfer mode | •Single transfer | Transfer is completed when the DMAi transfer counter (i = 0–1) underflows after reaching the terminal count. |
| | •Repeat transfer | When the DMAi transfer counter underflows, it is reloaded with the value of the DMAi transfer counter reload register and a DMA transfer is continued with it. |
| DMA interrupt request generation timing | | When the DMAi transfer counter underflowed |
| DMA startup | | Data transfer is initiated each time a DMA request is generated when the DMAiCON register's DMAE bit = "1" (enabled). |
| DMA shutdown | •Single transfer | • When the DMAE bit is set to "0" (disabled)<br>• After the DMAi transfer counter underflows |
| | •Repeat transfer | When the DMAE bit is set to "0" (disabled) |
| Reload timing for forward address pointer and transfer counter | | When a data transfer is started after setting the DMAE bit to "1" (enabled), the forward address pointer is reloaded with the value of the SARi or the DARi pointer whichever is specified to be in the forward direction and the DMAi transfer counter is reloaded with the value of the DMAi transfer counter reload register. |

Notes:

1. DMA transfer is not effective to any interrupt. DMA transfer is affected neither by the I flag nor by the interrupt control register.
2. The selectable causes of DMA requests differ with each channel.
3. Make sure that no DMAC-related registers (addresses $0020_{16}$ to $003F_{16}$) are accessed by the DMAC.

DMA0 request cause select register




Symbol: DM0SL | Address: 03B8$_{16}$ | After reset: 00$_{16}$

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| DSEL0 | DMA request cause select bit | Refer to note | RW |
| DSEL1 | | | RW |
| DSEL2 | | | RW |
| DSEL3 | | | RW |
| — (b5-b4) | Nothing is assigned. When write, set to “0”. When read, its content is “0”. | | — |
| DMS | DMA request cause expansion select bit | 0: Basic cause of request<br>1: Extended cause of request | RW |
| DSR | Software DMA request bit | A DMA request is generated by setting this bit to “1” when the DMS bit is “0” (basic cause) and the DSEL3 to DSEL0 bits are “0001$_2$” (software trigger).<br>The value of this bit when read is “0” . | RW |

Note 1: The causes of DMA0 requests can be selected by a combination of DMS bit and DSEL3 to DSEL0 bits in the manner described below.

| DSEL3 to DSEL0 | DMS=0 (basic cause of request) | DMS=1 (extended cause of request) |
|---|---|---|
| 0 0 0 0$_2$ | Falling edge of INT0 pin | – |
| 0 0 0 1$_2$ | Software trigger | – |
| 0 0 1 0$_2$ | Timer A0 | – |
| 0 0 1 1$_2$ | Timer A1 | – |
| 0 1 0 0$_2$ | Timer A2 | – |
| 0 1 0 1$_2$ | Timer A3 | – |
| 0 1 1 0$_2$ | Timer A4 (Note 4) | Two edges of INT0 pin |
| 0 1 1 1$_2$ | Timer B0 | Timer B3 |
| 1 0 0 0$_2$ | Timer B1 | Timer B4 |
| 1 0 0 1$_2$ | Timer B2 (Note 3) | Timer B5 |
| 1 0 1 0$_2$ | UART0 transmit | – |
| 1 0 1 1$_2$ | UART0 receive | – |
| 1 1 0 0$_2$ | UART2 transmit | – |
| 1 1 0 1$_2$ | UART2 receive | – |
| 1 1 1 0$_2$ | A-D conversion | – |
| 1 1 1 1$_2$ | UART1 transmit | – |

Note 2: In VINTi, INTRMTi, and HINTi (i = 0 to 3) of address 36$_{16}$ expansion register of expansion function, when use them by the following setup, DMA request cause extension select bit = "1" (extended cause of request) cannot be used.
- VINTi = 1011$_2$
- INTRMTi = 1010$_2$
- HINTi = 1001$_2$

(i = 0 to 3)

Note 3: Please change SECINTi (i = 0 to 3) in address 36$_{16}$ expansion registers of the expansion feature to the following settings when you use the DMA forwarding by timer B2 interrupt request.
- SECINTi = 0000$_2$

The DMA forwarding by the clock timer interrupt request cannot be used.

Note 4: Please change EXTIICINTi (i = 0 to 3) in address 02D6$_{16}$ I$^2$C0 interrupt control register to the following settings when you use the DMA forwarding by timer A4 interrupt request.
- EXTIICINTi = 0000$_2$

The DMA forwarding by the Multi-master I$^2$C interrupt request cannot be used.

**Figure 8.2 DM0SL Register**

DMA1 request cause select register




| Symbol | Address | After reset |
|---|---|---|
| DM1SL | 03BA16 | 0016 |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| DSEL0 | DMA request cause select bit | Refer to note | RW |
| DSEL1 | | | RW |
| DSEL2 | | | RW |
| DSEL3 | | | RW |
| —— (b5-b4) | Nothing is assigned. When write, set to “0”. When read, its content is “0”. | | —— |
| DMS | DMA request cause expansion select bit | 0: Basic cause of request<br>1: Extended cause of request | RW |
| DSR | Software DMA request bit | A DMA request is generated by setting this bit to “1” when the DMS bit is “0” (basic cause) and the DSEL3 to DSEL0 bits are “00012” (software trigger).<br>The value of this bit when read is “0” . | RW |

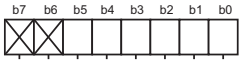
Note 1: The causes of DMA1 requests can be selected by a combination of DMS bit and DSEL3 to DSEL0 bits in the manner described below.

| DSEL3 to DSEL0 | DMS=0(basic cause of request) | DMS=1(extended cause of request) |
|---|---|---|
| 0 0 0 02 | Falling edge of INT1 pin | – |
| 0 0 0 12 | Software trigger | – |
| 0 0 1 02 | Timer A0 | – |
| 0 0 1 12 | Timer A1 | – |
| 0 1 0 02 | Timer A2 | – |
| 0 1 0 12 | Timer A3 | SI/O3 |
| 0 1 1 02 | Timer A4 (Note 3) | SI/O4 |
| 0 1 1 12 | Timer B0 | Two edges of $\overline{\text{INT1}}$ |
| 1 0 0 02 | Timer B1 | – |
| 1 0 0 12 | Timer B2 (Note 2) | – |
| 1 0 1 02 | UART0 transmit | – |
| 1 0 1 12 | UART0 receive/ACK0 | – |
| 1 1 0 02 | UART2 transmit | – |
| 1 1 0 12 | UART2 receive/ACK2 | – |
| 1 1 1 02 | A/D conversion | – |
| 1 1 1 12 | UART1 receive/ACK1 | – |

Note 2: Please change SECINTi (i = 0 to 3) in address 3616 enhancing registers of the enhanced feature to the following settings when you use the DMA forwarding by timer B2 interrupt request.
• SECINTi = 00002
The DMA forwarding by the clock timer interrupt request cannot be used.

Note 3: Please change EXTIICINTi (i = 0 to 3) in address 02D616 I2C0 interrupt control register to the following settings when you use the DMA forwarding by timer A4 interrupt request.
• EXTIICINTi = 00002
The DMA forwarding by the Multi-master I2C interrupt request cannot be used.

DMAi control register (i = 0,1)



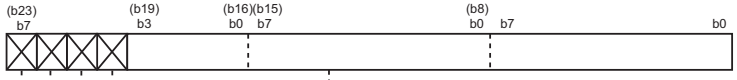

| Symbol | Address | After reset |
|---|---|---|
| DM0CON | 002C16 | 00000X002 |
| DM1CON | 003C16 | 00000X002 |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| DMBIT | Transfer unit bit select bit | 0 : 16 bits<br>1 : 8 bits | RW |
| DMASL | Repeat transfer mode select bit | 0 : Single transfer<br>1 : Repeat transfer | RW |
| DMAS | DMA request bit | 0 : DMA not requested<br>1 : DMA requested | RW<br>(Note 1) |
| DMAE | DMA enable bit | 0 : Disabled<br>1 : Enabled | RW |
| DSD | Source address direction select bit (Note 2) | 0 : Fixed<br>1 : Forward | RW |
| DAD | Destination address direction select bit (Note 2) | 0 : Fixed<br>1 : Forward | RW |
| —— (b7-b6) | Nothing is assigned. When write, set to “0”. When read, its content is “0”. | | —— |

Note 1: The DMAS bit can be set to “0” by writing “0” in a program (This bit remains unchanged even if “1” is written).
Note 2: At least one of the DAD and DSD bits must be “0” (address direction fixed).

**Figure 8.3 DM1SL Register, DM0CON Register, and DM1CON Registers**

DMAi source pointer (i = 0, 1) (Note)

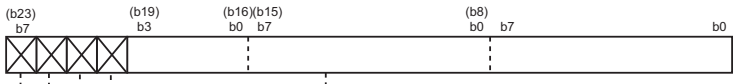



| Symbol | Address | After reset |
|---|---|---|
| SAR0 | 002216 to 002016 | Indeterminate |
| SAR1 | 003216 to 003016 | Indeterminate |

| Function | Setting range | RW |
|---|---|---|
| Set the source address of transfer | 0000016 to FFFFF16 | RW |
| Nothing is assigned. When write, set "0". When read, these contents are "0". | | — |

Note: If the DSD bit of DMiCON register is "0" (fixed), this register can only be written to when the DMAE bit of DMiCON register is "0" (DMA disabled).
If the DSD bit is "1" (forward direction), this register can be written to at any time.
If the DSD bit is "1" and the DMAE bit is "1" (DMA enabled), the DMAi forward address pointer can be read from this register. Otherwise, the value written to it can be read.

DMAi destination pointer (i = 0, 1)(Note)

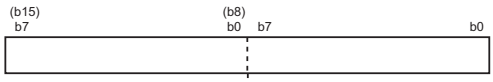



| Symbol | Address | After reset |
|---|---|---|
| DAR0 | 002616 to 002416 | Indeterminate |
| DAR1 | 003616 to 003416 | Indeterminate |

| Function | Setting range | RW |
|---|---|---|
| Set the destination address of transfer | 0000016 to FFFFF16 | RW |
| Nothing is assigned. When write, set "0". When read, these contents are "0". | | — |

Note: If the DAD bit of DMiCON register is "0" (fixed), this register can only be written to when the DMAE bit of DMiCON register is "0"(DMA disabled).
If the DAD bit is "1" (forward direction), this register can be written to at any time.
If the DAD bit is "1" and the DMAE bit is "1" (DMA enabled), the DMAi forward address pointer can be read from this register. Otherwise, the value written to it can be read.

DMAi transfer counter (i = 0, 1)




| Symbol | Address | After reset |
|---|---|---|
| TCR0 | 002916, 002816 | Indeterminate |
| TCR1 | 003916, 003816 | Indeterminate |

| Function | Setting range | RW |
|---|---|---|
| Set the transfer count minus 1. The written value is stored in the DMAi transfer counter reload register, and when the DMAE bit of DMiCON register is set to "1" (DMA enabled) or the DMAi transfer counter underflows when the DMASL bit of DMiCON register is "1" (repeat transfer), the value of the DMAi transfer counter reload register is transferred to the DMAi transfer counter. When read, the DMAi transfer counter is read. | 0000016 to FFFF16 | RW |

**Figure 8.4 SAR0, SAR1, DAR0, DAR1, TCR0, and TCR1 Registers**

## 8.1 Transfer Cycles

The transfer cycle consists of a memory or SFR read (source read) bus cycle and a write (destination write) bus cycle. The number of read and write bus cycles is affected by the source and destination addresses of transfer.

### (a) Effect of Source and Destination Addresses

If the transfer unit and data bus both are 16 bits and the source address of transfer begins with an odd address, the source read cycle consists of one more bus cycle than when the source address of transfer begins with an even address.

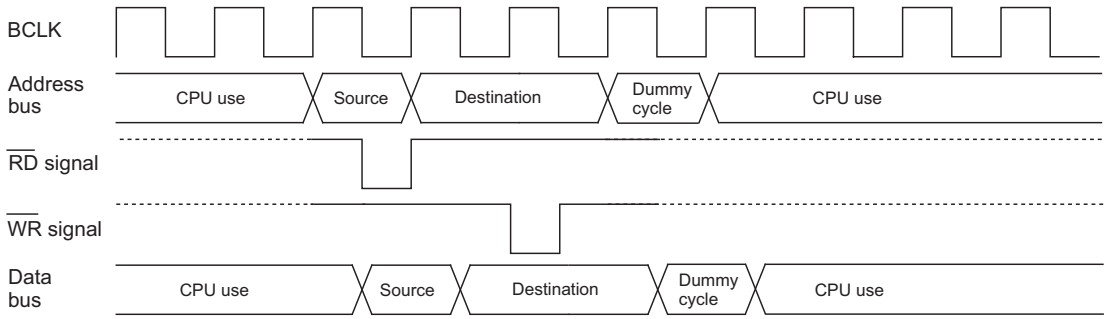
Similarly, if the transfer unit and data bus both are 16 bits and the destination address of transfer begins with an odd address, the destination write cycle consists of one more bus cycle than when the destination address of transfer begins with an even address.

### (b) Effect of Software Wait

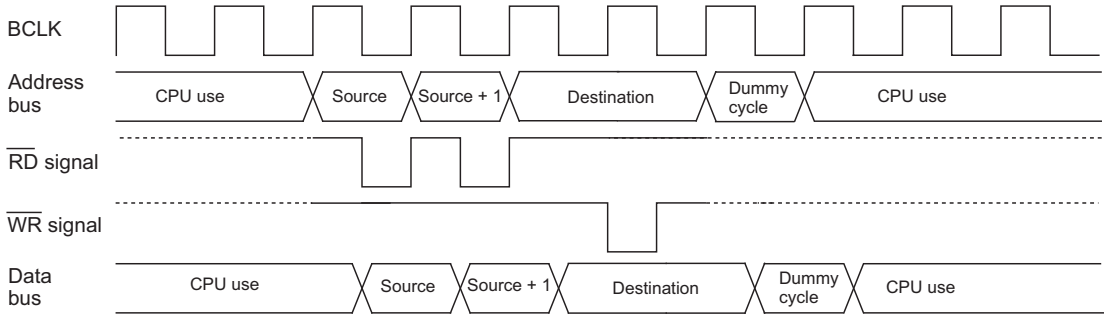
For memory or SFR accesses in which one or more software wait states are inserted, the number of bus cycles required for that access increases by an amount equal to software wait states.

Figure 8.5 shows the example of the cycles for a source read. For convenience, the destination write cycle is shown as one cycle and the source read cycles for the different conditions are shown. In reality, the destination write cycle is subject to the same conditions as the source read cycle, with the transfer cycle changing accordingly. When calculating transfer cycles, take into consideration each condition for the source read and the destination write cycle, respectively. For example, when data is transferred in 16 bit units using an 8-bit bus ((2) in Figure 8.5), two source read bus cycles and two destination write bus cycles are required.

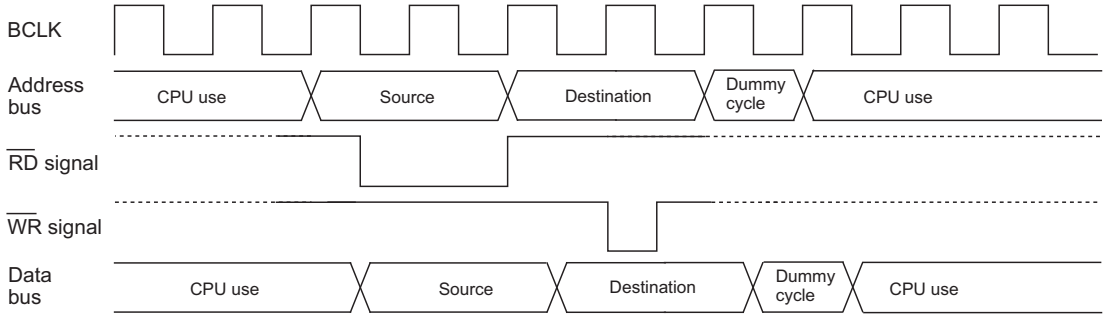
(1) When the transfer unit is 8 or 16 bits and the source of transfer is an even address



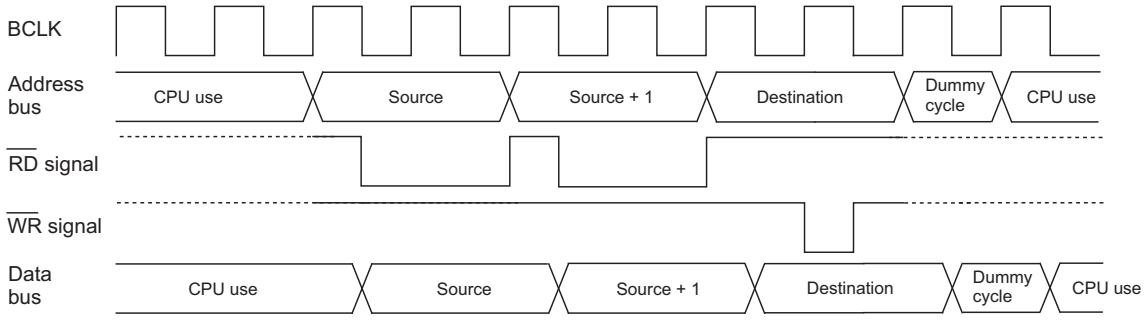

(2) When the transfer unit is 16 bits and the source address of transfer is an odd address, or when the transfer unit is 16 bits and an 8-bit bus is used




(3) When the source read cycle under condition (1) has one wait state inserted




(4) When the source read cycle under condition (2) has one wait state inserted




Note: The same timing changes occur with the respective conditions at the destination as at the source

**Figure 8.5 Transfer Cycles for Source Read**

## 8.2 Number of DMA Transfer Cycles

Any combination of even or odd transfer read and write addresses is possible. Table 8.2 shows the number of DMA transfer cycles. Table 8.3 shows the Coefficient j, k.
The number of DMAC transfer cycles can be calculated as follows:

No. of transfer cycles per transfer unit = No. of read cycles x j + No. of write cycles x k

**Table 8.2 Number of DMA Transfer Cycles**

| Transfer unit | Bus width | Access address | Single-chip mode | |
|---|---|---|---|---|
| | | | No. of read cycles | No. of write cycles |
| 8-bit transfers (DMBIT= “1”) | 16-bit (BYTE= “L”) | Even | 1 | 1 |
| | | Odd | 1 | 1 |
| 16-bit transfers (DMBIT= “0”) | 16-bit (BYTE = “L”) | Even | 1 | 1 |
| | | Odd | 2 | 2 |

**Table 8.3 Coefficient j, k**

| | Internal area | | |
|---|---|---|---|
| | Internal ROM, RAM | | SFR |
| | No wait | With wait | |
| j | 1 | 2 | 2 |
| k | 1 | 2 | 2 |

## 8.3 DMA Enable

When a data transfer starts after setting the DMAE bit in DMiCON register (i = 0, 1) to "1" (enabled), the DMAC operates as follows:

(1) Reload the forward address pointer with the SARi register value when the DSD bit in DMiCON register is "1" (forward) or the DARi register value when the DAD bit of DMiCON register is "1" (forward).
(2) Reload the DMAi transfer counter with the DMAi transfer counter reload register value.

If the DMAE bit is set to "1" again while it remains set, the DMAC performs the above operation. However, if a DMA request may occur simultaneously when the DMAE bit is being written, follow the steps below.
Step 1: Write "1" to the DMAE bit and DMAS bit in DMiCON register simultaneously.
Step 2: Make sure that the DMAi is in an initial state as described above (1) and (2) in a program.
If the DMAi is not in an initial state, the above steps should be repeated.

## 8.4 DMA Request

The DMAC can generate a DMA request as triggered by the cause of request that is selected with the DMS and DSEL3 to DSEL0 bits of DMiSL register (i = 0, 1) on either channel. Table 8.4 shows the timing at which the DMAS bit changes state.
Whenever a DMA request is generated, the DMAS bit is set to "1" (DMA requested) regardless of whether or not the DMAE bit is set. If the DMAE bit was set to "1" (enabled) when this occurred, the DMAS bit is set to "0" (DMA not requested) immediately before a data transfer starts. This bit cannot be set to "1" in a program (it can only be set to "0").
The DMAS bit may be set to "1" when the DMS or the DSEL3 to DSEL0 bits change state. Therefore, always be sure to set the DMAS bit to "0" after changing the DMS or the DSEL3 to DSEL0 bits.
Because if the DMAE bit is "1", a data transfer starts immediately after a DMA request is generated, the DMAS bit in almost all cases is "0" when read in a program. Read the DMAE bit to determine whether the DMAC is enabled.

**Table 8.4 Timing at Which the DMAS Bit Changes State**

| DMA factor | DMAS bit of the DMiCON register | |
|---|---|---|
| | Timing at which the bit is set to "1" | Timing at which the bit is set to "0" |
| Software trigger | When the DSR bit of DMiSL register is set to "1" | • Immediately before a data transfer starts<br>• When set by writing "0" in a program |
| Peripheral function | When the interrupt control register for the peripheral function that is selected by the DSEL3 to DSEL0 and DMS bits of DMiSL register has its IR bit set to "1" | |

## 8.5 Channel Priority and DMA Transfer Timing

If both DMA0 and DMA1 are enabled and DMA transfer request signals from DMA0 and DMA1 are detected active in the same sampling period (one period from a falling edge to the next falling edge of BCLK), the DMAS bit on each channel is set to "1" (DMA requested) at the same time. In this case, the DMA requests are arbitrated according to the channel priority, DMA0 > DMA1. The following describes DMAC operation when DMA0 and DMA1 requests are detected active in the same sampling period.

Figure 8.6 shows an example of DMA transfer effected by external factors.

DMA0 request having priority is received first to start a transfer when a DMA0 request and DMA1 request are generated simultaneously. After one DMA0 transfer is completed, a bus arbitration is returned to the CPU. When the CPU has completed one bus access, a DMA1 transfer starts. After one DMA1 transfer is completed, the bus arbitration is again returned to the CPU.

In addition, DMA requests cannot be counted up since each channel has one DMAS bit. Therefore, when DMA requests, as DMA1 in Figure 8.6, occurs more than one time, the DMAS bit is set to "0" as soon as getting the bus arbitration. The bus arbitration is returned to the CPU when one transfer is completed.

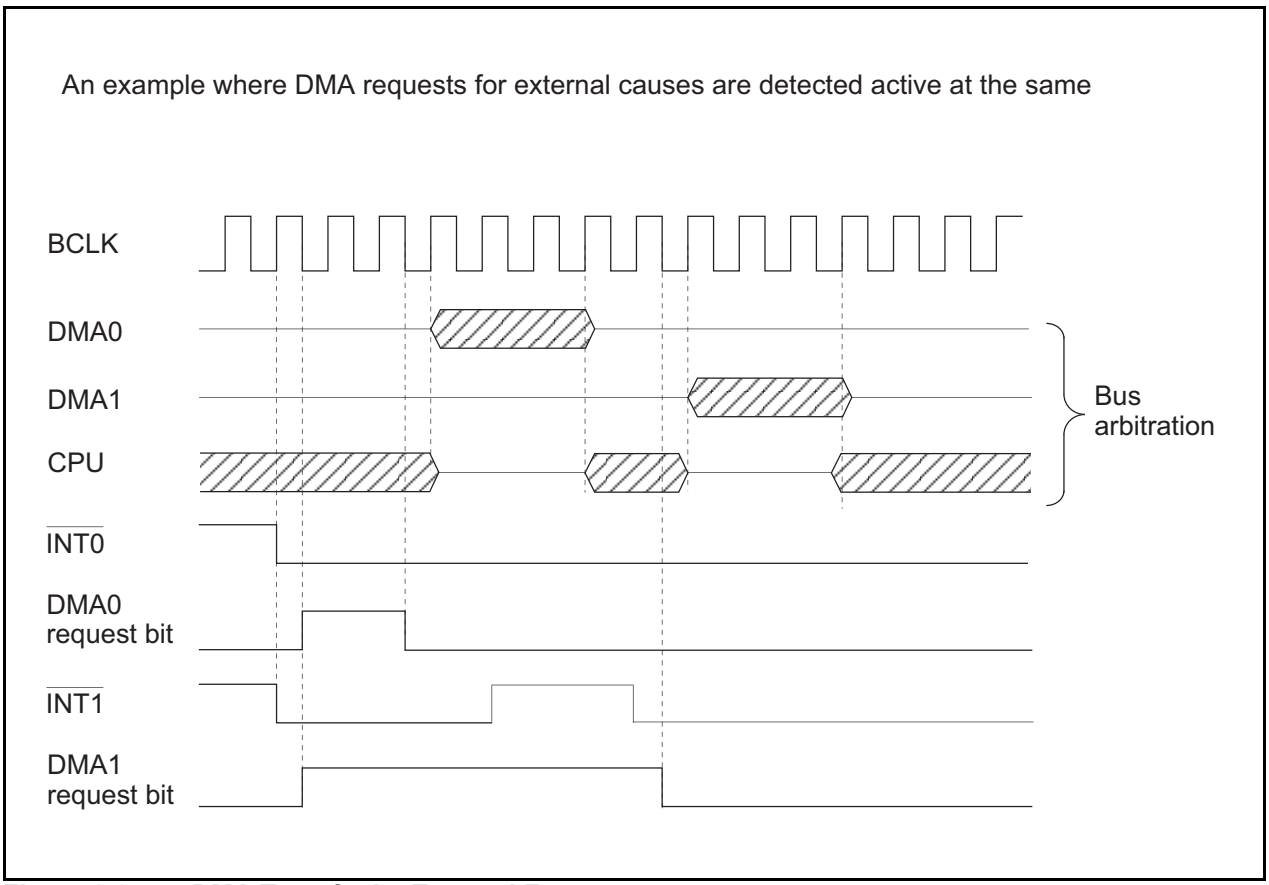
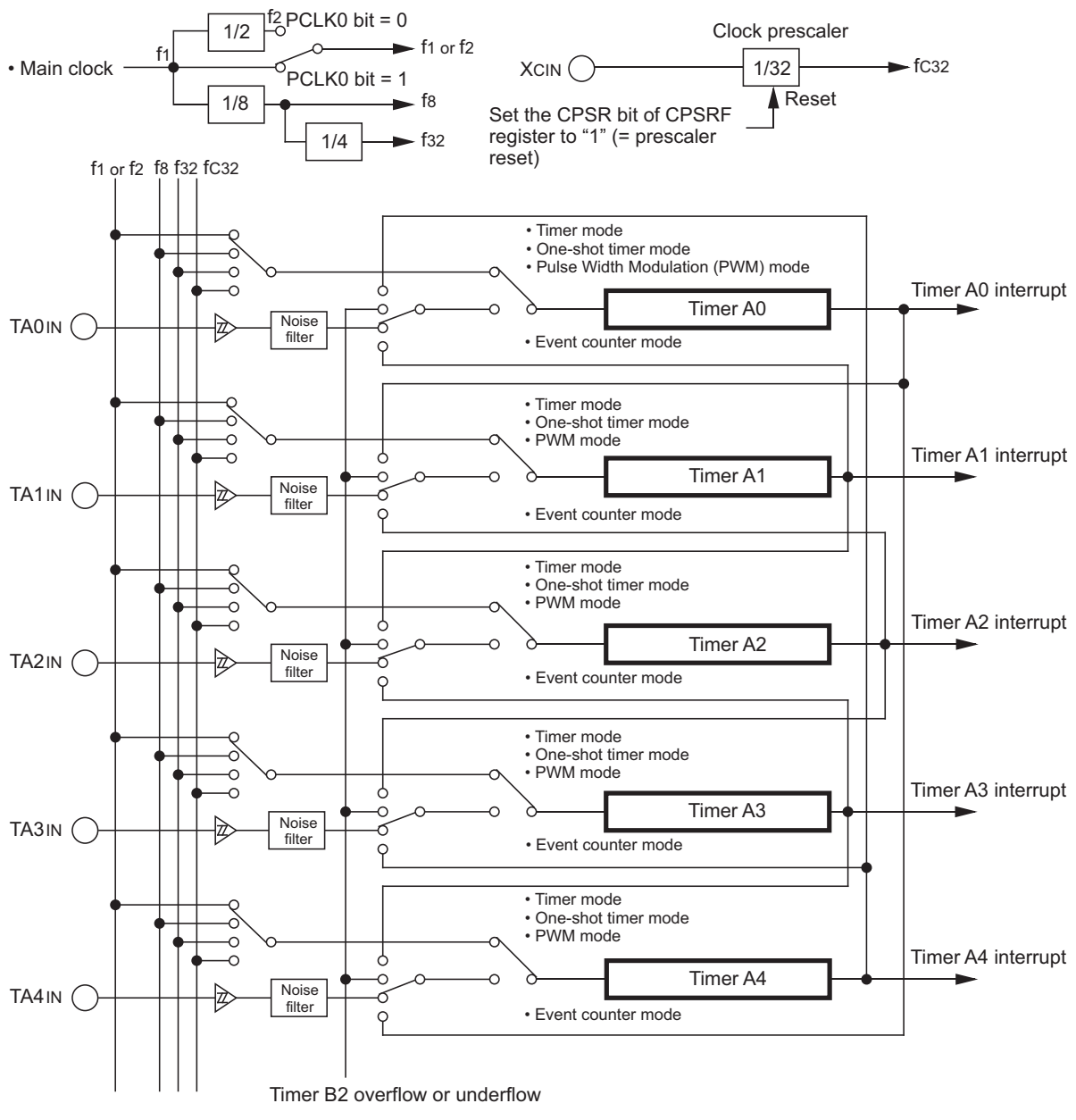



**Figure 8.6 DMA Transfer by External Factors**

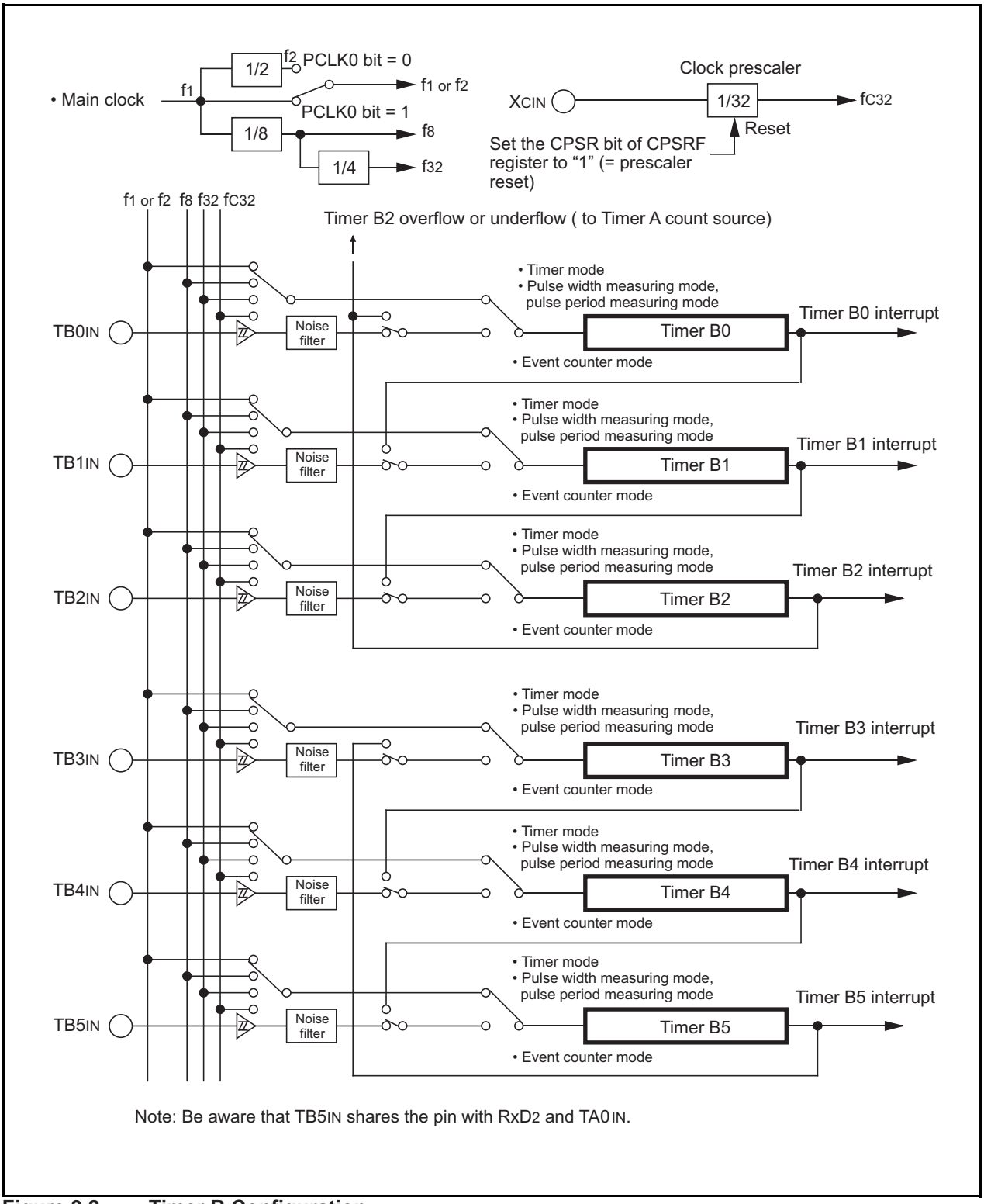
# 9. Timers

Eleven 16-bit timers, each capable of operating independently of the others, can be classified by function as either timer A (five) and timer B (six). The count source for each timer acts as a clock, to control such timer operations as counting, reloading, etc. Figures 9.1 and 9.2 show block diagrams of timer A and timer B configuration, respectively.




Note: Be aware that TA0IN shares the pin with RxD2 and TB5IN.

**Figure 9.1 Timer A Configuration**



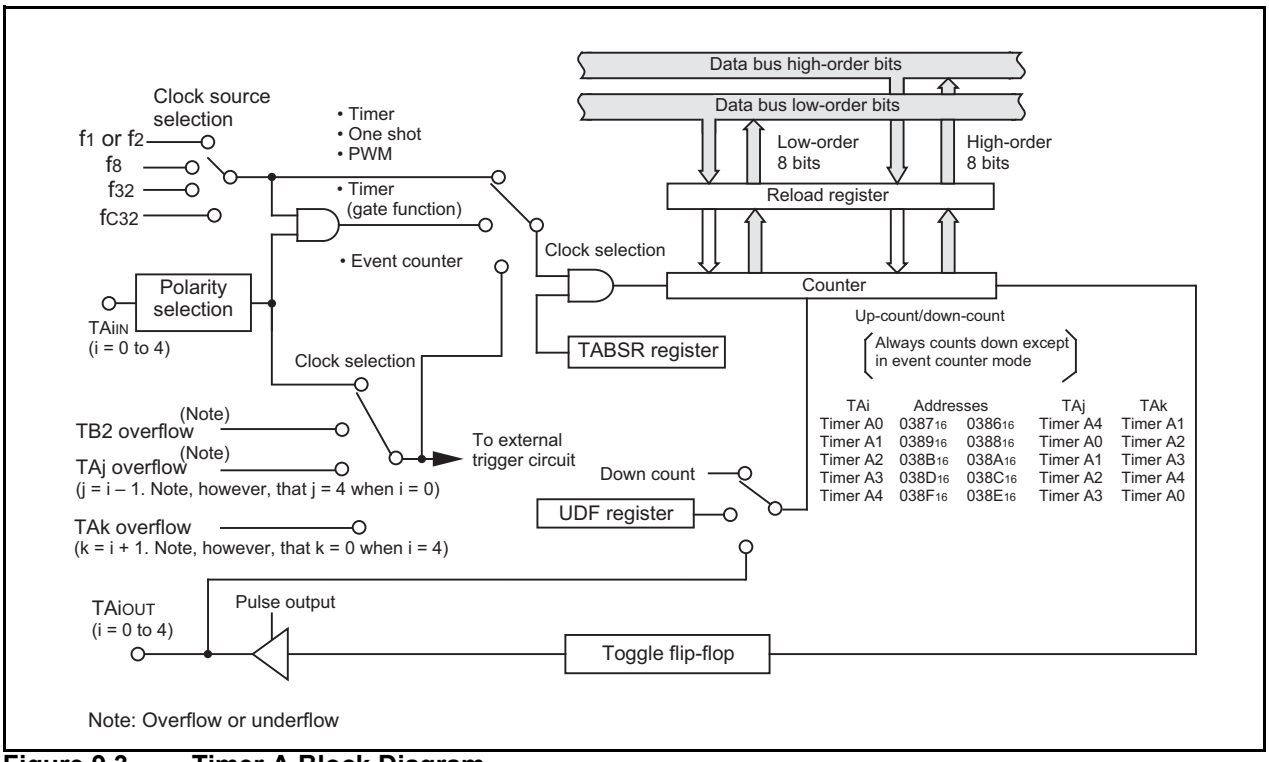

Note: Be aware that TB5IN shares the pin with RxD2 and TA0IN.

**Figure 9.2 Timer B Configuration**

## 9.1 Timer A

Figure 9.3 shows a block diagram of the timer A. Figures 9.4 to 9.6 show registers related to the timer A.
The timer A supports the following four modes. Except in event counter mode, timers A0 to A4 all have the same function. Use the TMOD1 to TMOD0 bits of TAiMR register (i = 0 to 4) to select the desired mode.

- Timer mode: The timer counts an internal count source.
- Event counter mode: The timer counts pulses from an external device or overflows and underflows of other timers.
- One-shot timer mode: The timer outputs a pulse only once before it reaches the minimum count "0000$_{16}$."
- Pulse width modulation (PWM) mode: The timer outputs pulses in a given width successively.




| TAi | Addresses | | TAj | TAk |
|---|---|---|---|---|
| Timer A0 | 0387$_{16}$ | 0386$_{16}$ | Timer A4 | Timer A1 |
| Timer A1 | 0389$_{16}$ | 0388$_{16}$ | Timer A0 | Timer A2 |
| Timer A2 | 038B$_{16}$ | 038A$_{16}$ | Timer A1 | Timer A3 |
| Timer A3 | 038D$_{16}$ | 038C$_{16}$ | Timer A2 | Timer A4 |
| Timer A4 | 038F$_{16}$ | 038E$_{16}$ | Timer A3 | Timer A0 |

**Figure 9.3 Timer A Block Diagram**

Timer Ai mode register (i=0 to 4)

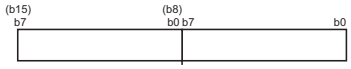
| Symbol | Address | After reset |
|---|---|---|
| TA0MR to TA4MR | 0396$_{16}$ to 039A$_{16}$ | 00$_{16}$ |

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 0 : Timer mode<br>0 1 : Event counter mode<br>1 0 : One-shot timer mode<br>1 1 : Pulse width modulation (PWM) mode | RW |
| TMOD1 | | | RW |
| MR0 | ——— | Function varies with each operation mode | RW |
| MR1 | | | RW |
| MR2 | | | RW |
| MR3 | | | RW |
| TCK0 | Count source select bit | Function varies with each operation mode | RW |
| TCK1 | | | RW |

**Figure 9.4 TA0MR to TA4MR Registers**

Timer Ai register (i= 0 to 4) (Note 1)




| Symbol | Address | After reset |
|---|---|---|
| TA0 | 0387$_{16}$, 0386$_{16}$ | Indeterminate |
| TA1 | 0389$_{16}$, 0388$_{16}$ | Indeterminate |
| TA2 | 038B$_{16}$, 038A$_{16}$ | Indeterminate |
| TA3 | 038D$_{16}$, 038C$_{16}$ | Indeterminate |
| TA4 | 038F$_{16}$, 038E$_{16}$ | Indeterminate |

| Mode | Function | Setting range | RW |
|---|---|---|---|
| Timer mode | Divide the count source by n + 1 where n = set value | 0000$_{16}$ to FFFF$_{16}$ | RW |
| Event counter mode | Divide the count source by FFFF$_{16}$ – n + 1 where n = set value when counting up or by n + 1 when counting down (Note 5) | 0000$_{16}$ to FFFF$_{16}$ | RW |
| One-shot timer mode | Divide the count source by n where n = set value and cause the timer to stop | 0000$_{16}$ to FFFF$_{16}$ (Notes 2, 4) | WO |
| Pulse width modulation mode (16-bit PWM) | Modify the pulse width as follows:<br>PWM period: ($2^{16}$ – 1) / fj<br>High level PWM pulse width: n / fj<br>where n = set value, fj = count source frequency | 0000$_{16}$ to FFFE$_{16}$ (Note 3, 4) | WO |
| Pulse width modulation mode (8-bit PWM) | Modify the pulse width as follows:<br>PWM period: ($2^8$ – 1) x (m + 1)/ fj<br>High level PWM pulse width: (m + 1)n / fj<br>where n = high-order address set value, m = low-order address set value, fj = count source frequency | 00$_{16}$ to FE$_{16}$ (High-order address)<br>00$_{16}$ to FF$_{16}$ (Low-order address)<br>(Note 3, 4) | WO |

Note 1: The register must be accessed in 16 bit units.
Note 2: If the TAi register is set to '0000$_{16}$,' the counter does not work and timer Ai interrupt requests are not generated either. Furthermore, if "pulse output" is selected, no pulses are output from the TAiOUT pin.
Note 3: If the TAi register is set to '0000$_{16}$,' the pulse width modulator does not work, the output level on the TAiOUT pin remains low, and timer Ai interrupt requests are not generated either. The same applies when the 8 high-order bits of the timer TAi register are set to '001 6' while operating as an 8-bit pulse width modulator.
Note 4: Use the MOV instruction to write to the TAi register.
Note 5: The timer counts pulses from an external device or overflows or underflows in other timers.

Count start flag




| Symbol | Address | After reset |
|---|---|---|
| TABSR | 0380$_{16}$ | 00$_{16}$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| TA0S | Timer A0 count start flag | 0 : Stops counting<br>1 : Starts counting | RW |
| TA1S | Timer A1 count start flag | | RW |
| TA2S | Timer A2 count start flag | | RW |
| TA3S | Timer A3 count start flag | | RW |
| TA4S | Timer A4 count start flag | | RW |
| TB0S | Timer B0 count start flag | | RW |
| TB1S | Timer B1 count start flag | | RW |
| TB2S | Timer B2 count start flag | | RW |

Up/down flag (Note 1)




| Symbol | Address | After reset |
|---|---|---|
| UDF | 0384$_{16}$ | 00$_{16}$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| TA0UD | Timer A0 up/down flag | 0 : Down count<br>1 : Up count<br>Enabled by setting the TAiMR register's MR2 bit to "0" (= switching source in UDF register) during event counter mode. | RW |
| TA1UD | Timer A1 up/down flag | | RW |
| TA2UD | Timer A2 up/down flag | | RW |
| TA3UD | Timer A3 up/down flag | | RW |
| TA4UD | Timer A4 up/down flag | | RW |
| TA2P | Timer A2 two-phase pulse signal processing select bit | 0 : two-phase pulse signal processing disabled<br>1 : two-phase pulse signal processing enabled<br>(Notes 2, 3) | WO |
| TA3P | Timer A3 two-phase pulse signal processing select bit | | WO |
| TA4P | Timer A4 two-phase pulse signal processing select bit | | WO |

Note 1: Use MOV instruction to write to this register.
Note 2: Make sure the port direction bits for the TA2IN to TA4IN and TA2OUT to TA4OUT pins are set to "0" (input mode).
Note 3: When not using the two-phase pulse signal processing function, set the bit corresponding to timer A2 to timer A4 to "0"

**Figure 9.5 TA0 to TA4 Registers, TABSR Register, and UDF Register**

### One-shot start flag

| Symbol | Address | After reset |
|---|---|---|
| ONSF | 038216 | 0016 |

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | 0 | | | | | |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| TA0OS | Timer A0 one-shot start flag | The timer starts counting by setting this bit to "1" while the TMOD1 to TMOD0 bits of TAiMR register (i = 0 to 4) = '102' (= one-shot timer mode) and the MR2 bit of TAiMR register = "0" (=TAiOS bit enabled). When read, its content is "0". | RW |
| TA1OS | Timer A1 one-shot start flag | | RW |
| TA2OS | Timer A2 one-shot start flag | | RW |
| TA3OS | Timer A3 one-shot start flag | | RW |
| TA4OS | Timer A4 one-shot start flag | | RW |
| (b5) | Reserved bit | Must be set to "0" | RW |
| TA0TGL | Timer A0 event/trigger select bit | b7 b6<br>0 0 : Input on TA0IN is selected (Note 1)<br>0 1 : TB2 overflow is selected (Note 2)<br>1 0 : TA4 overflow is selected (Note 2)<br>1 1 : TA1 overflow is selected (Note 2) | RW |
| TA0TGH | | | RW |

Note 1: Make sure the PD7_1 bit of PD7 register is set to "0" (= input mode).
Note 2: Overflow or underflow

### Trigger select register

| Symbol | Address | After reset |
|---|---|---|
| TRGSR | 038316 | 0016 |

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| TA1TGL | Timer A1 event/trigger select bit | b1 b0<br>0 0 : Input on TA1IN is selected (Note 1)<br>0 1 : TB2 is selected<br>1 0 : TA0 is selected<br>1 1 : TA2 is selected | RW |
| TA1TGH | | | RW |
| TA2TGL | Timer A2 event/trigger select bit | b3 b2<br>0 0 : Input on TA2IN is selected (Note 1)<br>0 1 : TB2 is selected<br>1 0 : TA1 is selected<br>1 1 : TA3 is selected | RW |
| TA2TGH | | | RW |
| TA3TGL | Timer A3 event/trigger select bit | b5 b4<br>0 0 : Input on TA3IN is selected (Note 1)<br>0 1 : TB2 is selected<br>1 0 : TA2 is selected<br>1 1 : TA4 is selected | RW |
| TA3TGH | | | RW |
| TA4TGL | Timer A4 event/trigger select bit | b7 b6<br>0 0 : Input on TA4IN is selected (Note 1)<br>0 1 : TB2 is selected<br>1 0 : TA3 is selected<br>1 1 : TA0 is selected | RW |
| TA4TGH | | | RW |

Note 1: Make sure the port direction bits for the TA1IN to TA4IN pins are set to "0" (= input mode).

### Clock prescaler reset flag

| Symbol | Address | After reset |
|---|---|---|
| CPSRF | 038116 | 0XXXXXXX2 |

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| (b6-b0) | Nothing is assigned. When write, set to "0". When read, their contents are indeterminate. | | — |
| CPSR | Clock prescaler reset flag | Setting this bit to "1" initializes the prescaler for the timekeeping clock. (When read, its content is "0".) | RW |

**Figure 9.6 ONSF Register, TRGSR Register, and CPSRF Register**

### 9.1.1 Timer Mode

In timer mode, the timer counts a count source generated internally (see Table 9.1). Figure 9.7 shows TAiMR register in timer mode.

**Table 9.1 Specifications in Timer Mode**

| Item | Specification |
|---|---|
| Count source | $f_1$, $f_2$, $f_8$, $f_{32}$, $f_{C32}$ |
| Count operation | • Down-count<br>• When the timer underflows, it reloads the reload register contents and continues counting |
| Divide ratio | 1/(n+1) n: set value of TAi register (i= 0 to 4) $0000_{16}$ to $FFFF_{16}$ |
| Count start condition | Set TAiS bit of TABSR register to "1" (= start counting) |
| Count stop condition | Set TAiS bit to "0" (= stop counting) |
| Interrupt request generation timing | Timer underflow |
| TAiIN pin function | I/O port or gate input |
| TAiOUT pin function | I/O port or pulse output |
| Read from timer | Count value can be read by reading TAi register |
| Write to timer | • When not counting and until the 1st count source is input after counting start<br>Value written to TAi register is written to both reload register and counter<br>• When counting (after 1st count source input)<br>Value written to TAi register is written to only reload register<br>(Transferred to counter when reloaded next) |
| Select function | • Gate function<br>Counting can be started and stopped by an input signal to TAiIN pin<br>• Pulse output function<br>Whenever the timer underflows, the output polarity of TAiOUT pin is inverted.<br>When not counting, the pin outputs a low. |

Timer Ai mode register (i=0 to 4)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | 0 | | | | 0 | 0 |

Symbol: TA0MR to TA4MR — Address: $0396_{16}$ to $039A_{16}$ — After reset: $00_{16}$

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 0 : Timer mode | RW |
| TMOD1 | | | RW |
| MR0 | Pulse output function select bit | 0 : Pulse is not output (TAiOUT pin is a normal port pin)<br>1 : Pulse is output (Note 1) (TAiOUT pin is a pulse output pin) | RW |
| MR1 | Gate function select bit | b4 b3<br>0 0 : } Gate function not available (TAiIN pin functions as I/O port)<br>0 1 :<br>1 0 : Counts while input on the TAiIN pin is low (Note 2)<br>1 1 : Counts while input on the TAiIN pin is high (Note 2) | RW |
| MR2 | | | RW |
| MR3 | Must be set to "0" in timer mode | | RW |
| TCK0 | Count source select bit | b7 b6<br>0 0 : $f_1$ or $f_2$<br>0 1 : $f_8$<br>1 0 : $f_{32}$<br>1 1 : $f_{C32}$ | RW |
| TCK1 | | | RW |

Note 1: TA0OUT pin is N-channel open drain output.
Note 2: The port direction bit for the TAiIN pin must be set to "0" (= input mode).

**Figure 9.7 Timer Ai Mode Register in Timer Mode**

### 9.1.2 Event Counter Mode

In event counter mode, the timer counts pulses from an external device or overflows and underflows of other timers. Timers A2, A3 and A4 can count two-phase external signals. Table 9.2 lists specifications in event counter mode (when not processing two-phase pulse signal). Table 9.3 lists specifications in event counter mode (when processing two-phase pulse signal with the timers A2, A3 and A4). Figure 9.8 shows TAiMR register in event counter mode (when not processing two-phase pulse signal). Figure 9.9 shows TA2MR to TA4MR registers in event counter mode (when processing two-phase pulse signal with the timers A2, A3 and A4).

**Table 9.2 Specifications in Event Counter Mode (when not processing two-phase pulse signal)**

| Item | Specification |
|---|---|
| Count source | • External signals input to TAiIN pin (i=0 to 4) (effective edge can be selected in program)<br>• Timer B2 overflows or underflows, timer Aj (j=i-1, except j=4 if i=0) overflows or underflows, timer Ak (k=i+1, except k=0 if i=4) overflows or underflows |
| Count operation | • Up-count or down-count can be selected by external signal or program<br>• When the timer overflows or underflows, it reloads the reload register contents and continues counting. When operating in free-running mode, the timer continues counting without reloading. |
| Divided ratio | 1/ ($FFFF_{16}$ - n + 1) for up-count<br>1/ (n + 1) for down-count n : set value of TAi register $0000_{16}$ to $FFFF_{16}$ |
| Count start condition | Set TAiS bit of TABSR register to “1” (= start counting) |
| Count stop condition | Set TAiS bit to “0” (= stop counting) |
| Interrupt request generation timing | Timer overflow or underflow |
| TAiIN pin function | I/O port or count source input |
| TAiOUT pin function | I/O port, pulse output, or up/down-count select input |
| Read from timer | Count value can be read by reading TAi register |
| Write to timer | • When not counting and until the 1st count source is input after counting start<br>Value written to TAi register is written to both reload register and counter<br>• When counting (after 1st count source input)<br>Value written to TAi register is written to only reload register<br>(Transferred to counter when reloaded next) |
| Select function | • Free-run count function<br>Even when the timer overflows or underflows, the reload register content is not reloaded to it<br>• Pulse output function<br>Whenever the timer overflows or underflows, the output polarity of TAiOUT pin is inverted . When not counting, the pin outputs a low. |

Timer Ai mode register (i=0 to 4)
(When not using two-phase pulse signal processing)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | 0 | | | | 0 | 1 |

| Symbol | Address | After reset |
|---|---|---|
| TA0MR to TA4MR | $0396_{16}$ to $039A_{16}$ | $00_{16}$ |

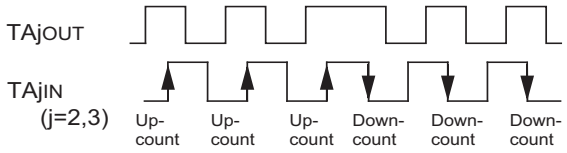
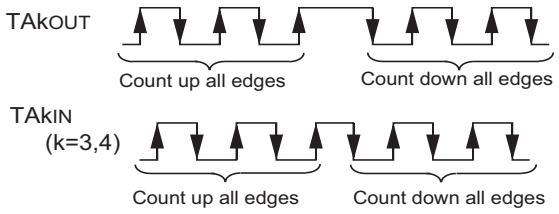
| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 1 : Event counter mode (Note 1) | RW |
| TMOD1 | | | RW |
| MR0 | Pulse output function select bit | 0 : Pulse is not output (TAiOUT pin functions as I/O port)<br>1 : Pulse is output (Note 2) (TAiOUT pin functions as pulse output pin) | RW |
| MR1 | Count polarity select bit (Note 3) | 0 : Counts external signal's falling edge<br>1 : Counts external signal's rising edge | RW |
| MR2 | Up/down switching cause select bit | 0 : UDF register<br>1 : Input signal to TAiOUT pin (Note 4) | RW |
| MR3 | Must be set to "0" in event counter mode | | RW |
| TCK0 | Count operation type select bit | 0 : Reload type<br>1 : Free-run type | RW |
| TCK1 | Can be "0" or "1" when not using two-phase pulse signal processing | | RW |

Note 1: During event counter mode, the count source can be selected using the ONSF and TRGSR registers.
Note 2: TA0OUT pin is N-channel open drain output.
Note 3: Effective when the TAiGH and TAiGL bits of ONSF or TRGSR register are '$00_2$' (TAiIN pin input).
Note 4: Count down when input on TAiOUT pin is low or count up when input on that pin is high. The port direction bit for TAiOUT pin must be set to "0" (= input mode).

**Figure 9.8 TAiMR Register in Event Counter Mode (when not using two-phase pulse signal processing)**

The use of the event counter mode (When you use two aspect pulse signal processing with Timer A2, A3, and A4) is shown in Table 9.3.
Figure 9.9 shows from TA2MR register to TA4MR register (When you use two aspect pulse signal processing with timer A2, A3, and A4) at event counter mode.

**Table 9.3 Specifications in Event Counter Mode (when processing two-phase pulse signal with timers A2, A3 and A4))**

| Item | Specification |
|---|---|
| Count source | • Two-phase pulse signals input to TAiIN or TAiOUT pins (i = 2 to 4) |
| Count operation | • Up-count or down-count can be selected by two-phase pulse signal<br>• When the timer overflows or underflows, it reloads the reload register contents and continues counting. When operating in free-running mode, the timer continues counting without reloading. |
| Divide ratio | 1/ (FFFF16 - n + 1) for up-count<br>1/ (n + 1) for down-count n : set value of TAi register 000016 to FFFF16 |
| Count start condition | Set TAiS bit of TABSR register to "1" (= start counting) |
| Count stop condition | Set TAiS bit to "0" (= stop counting) |
| Interrupt request generation timing | Timer overflow or underflow |
| TAiIN pin function | Two-phase pulse input |
| TAiOUT pin function | Two-phase pulse input |
| Read from timer | Count value can be read by reading timer A2, A3 or A4 register |
| Write to timer | • When not counting and until the 1st count source is input after counting start<br>Value written to TAi register is written to both reload register and counter<br>• When counting (after 1st count source input)<br>Value written to TAi register is written to reload register<br>(Transferred to counter when reloaded next) |
| Select function (Note) | • Normal processing operation (timer A2 and timer A3)<br>The timer counts up rising edges or counts down falling edges on TAjIN pin when input signals on TAjOUT pin is "H".<br><br>• Multiply-by-4 processing operation (timer A3 and timer A4)<br>If the phase relationship is such that TAkIN(k=3, 4) pin goes "H" when the input signal on TAkOUT pin is "H", the timer counts up rising and falling edges on TAkOUT and TAkIN pins. If the phase relationship is such that TAkIN pin goes "L" when the input signal on TAkOUT pin is "H", the timer counts down rising and falling edges on TAkOUT and TAkIN pins.<br> |

Notes:
1. Only timer A3 is selectable. Timer A2 is fixed to normal processing operation, and timer A4 is fixed to multiply-by-4 processing operation.

Timer Ai mode register (i=2 to 4)
(When using two-phase pulse signal processing)

| b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|
| | | 0 | 1 | 0 | 0 | 0 | 1 |

Symbol: TA2MR to TA4MR
Address: 039816 to 039A16
After reset: 0016

| | Bit name | Function | RW |
|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 1 : Event counter mode | RW |
| TMOD1 | | | RW |
| MR0 | To use two-phase pulse signal processing, set this bit to “0”. | | RW |
| MR1 | To use two-phase pulse signal processing, set this bit to “0”. | | RW |
| MR2 | To use two-phase pulse signal processing, set this bit to “1”. | | RW |
| MR3 | To use two-phase pulse signal processing, set this bit to “0”. | | RW |
| TCK0 | Count operation type select bit | 0 : Reload type<br>1 : Free-run type | RW |
| TCK1 | Two-phase pulse signal processing operation select bit (Note 1)(Note 2) | 0 : Normal processing operation<br>1 : Multiply-by-4 processing operation | RW |

Note 1: TCK1 bit is valid for timer A3 mode register. No matter how this bit is set, timers A2 and A4 always operate in normal processing mode and x4 processing mode, respectively.
Note 2: If two-phase pulse signal processing is desired, following register settings are required:
- Set the UDF register's TAiP bit to “1” (two-phase pulse signal processing function enabled) .
- Set the TRGSR register's TAiGH and TAiGL bits to “002” (TAiIN pin input).
- Set the port direction bits for TA iIN and TAiOUT to “0” (input mode).

**Figure 9.9 TA2MR to TA4MR Registers in Event Counter Mode (when using two-phase pulsesignal processing with timer A2, A3 or A4)**

### 9.1.3 One-shot Timer Mode

In one-shot timer mode, the timer is activated only once by one trigger. (See Table 9.4.) When the trigger occurs, the timer starts up and continues operating for a given period. Figure 9.10 shows the TAiMR register in one-shot timer mode.

**Table 9.4 Specifications in One-shot Timer Mode**

| Item | Specification |
|---|---|
| Count source | $f_1$, $f_2$, $f_8$, $f_{32}$, $f_{C32}$ |
| Count operation | • Down-count<br>• When the counter reaches $0000_{16}$, it stops counting after reloading a new value<br>• If a trigger occurs when counting, the timer reloads a new count and restarts counting |
| Divide ratio | 1/n n : set value of TAi register $0000_{16}$ to $FFFF_{16}$<br>However, the counter does not work if the divide-by-n value is set to $0000_{16}$. |
| Count start condition | TAiS bit of TABSR register = "1" (start counting) and one of the following triggers occurs.<br>• External trigger input from the TAiIN pin<br>• Timer B2 overflow or underflow,<br>timer Aj (j=i-1, except j=4 if i=0) overflow or underflow,<br>timer Ak (k=i+1, except k=0 if i=4) overflow or underflow<br>• The TAiOS bit of ONSF register is set to "1" (= timer starts) |
| Count stop condition | • When the counter is reloaded after reaching "$0000_{16}$"<br>• TAiS bit is set to "0" (= stop counting) |
| Interrupt request generation timing | When the counter reaches "$0000_{16}$" |
| TAiIN pin function | I/O port or trigger input |
| TAiOUT pin function | I/O port or pulse output |
| Read from timer | An indeterminate value is read by reading TAi register |
| Write to timer | • When not counting and until the 1st count source is input after counting start<br>Value written to TAi register is written to both reload register and counter<br>• When counting (after 1st count source input)<br>Value written to TAi register is written to only reload register<br>(Transferred to counter when reloaded next) |
| Select function | • Pulse output function<br>The timer outputs a low when not counting and a high when counting. |

Timer Ai mode register (i=0 to 4)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | 0 | | | | 1 | 0 |

| Symbol | Address | After reset |
|---|---|---|
| TA0MR to TA4MR | $0396_{16}$ to $039A_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>1 0 : One-shot timer mode | RW |
| TMOD1 | | | RW |
| MR0 | Pulse output function select bit | 0 : Pulse is not output (TAiOUT pin functions as I/O port)<br>1 : Pulse is output (Note 1) (TAiOUT pin functions as a pulse output pin) | RW |
| MR1 | External trigger select bit (Note 2) | 0 : Falling edge of input signal to TAiIN pin (Note 3)<br>1 : Rising edge of input signal to TAiIN pin (Note 3) | RW |
| MR2 | Trigger select bit | 0 : TAiOS bit is enabled<br>1 : Selected by TAiTGH to TAiTGL bits | RW |
| MR3 | Must be set to “0” in one-shot timer mode | | RW |
| TCK0 | Count source select bit | b7 b6<br>0 0 : $f_1$ or $f_2$<br>0 1 : $f_8$<br>1 0 : $f_{32}$<br>1 1 : $fC_{32}$ | RW |
| TCK1 | | | RW |

Note 1: TA0OUT pin is N-channel open drain output.
Note 2: Effective when the TAiTGH and TAiTGL bits of ONSF or TRGSR register are ‘$00_2$’ (TAiIN pin input).
Note 3: The port direction bit for the TAiIN pin must be set to “0” (= input mode).

**Figure 9.10 TAiMR Register in One-shot Timer Mode**

### 9.1.4 Pulse Width Modulation (PWM) Mode

In PWM mode, the timer outputs pulses of a given width in succession (see Table 9.5). The counter functions as either 16-bit pulse width modulator or 8-bit pulse width modulator. Figure 9.11 shows TAiMR register in pulse width modulation mode. Figures 9.12 and 9.13 show examples of how a 16-bit pulse width modulator operates and how an 8-bit pulse width modulator operates.

**Table 9.5 Specifications in PWM Mode**

| Item | Specification |
|---|---|
| Count source | $f_1$, $f_2$, $f_8$, $f_{32}$, $f_{C32}$ |
| Count operation | • Down-count (operating as an 8-bit or a 16-bit pulse width modulator)<br>• The timer reloads a new value at a rising edge of PWM pulse and continues counting<br>• The timer is not affected by a trigger that occurs during counting |
| 16-bit PWM | • High level width n / fj n : set value of TAi register (i=o to 4)<br>• Cycle time $(2^{16}-1)$ / fj fixed fj: count source frequency ($f_1$, $f_2$, $f_8$, $f_{32}$, $f_{C32}$) |
| 8-bit PWM | • High level width n x (m+1) / fj n : set value of TAi register high-order address<br>• Cycle time $(2^8-1)$ x (m+1) / fj m : set value of TAi register low-order address |
| Count start condition | • TAiS bit of TABSR register is set to “1” (= start counting)<br>• The TAiS bit = 1 and external trigger input from the $TA_{iIN}$ pin<br>• The TAiS bit = 1 and one of the following external triggers occurs<br>• Timer B2 overflow or underflow,<br>timer Aj (j=i-1, except j=4 if i=0) overflow or underflow,<br>timer Ak (k=i+1, except k=0 if i=4) overflow or underflow |
| Count stop condition | TAiS bit is set to “0” (= stop counting) |
| Interrupt request generation timing | PWM pulse goes “L” |
| $TA_{iIN}$ pin function | I/O port or trigger input |
| $TA_{iOUT}$ pin function | Pulse output |
| Read from timer | An indeterminate value is read by reading TAi register |
| Write to timer | • When not counting and until the 1st count source is input after counting start<br>Value written to TAi register is written to both reload register and counter<br>• When counting (after 1st count source input)<br>Value written to TAi register is written to only reload register<br>(Transferred to counter when reloaded next) |

Timer Ai mode register (i= 0 to 4)

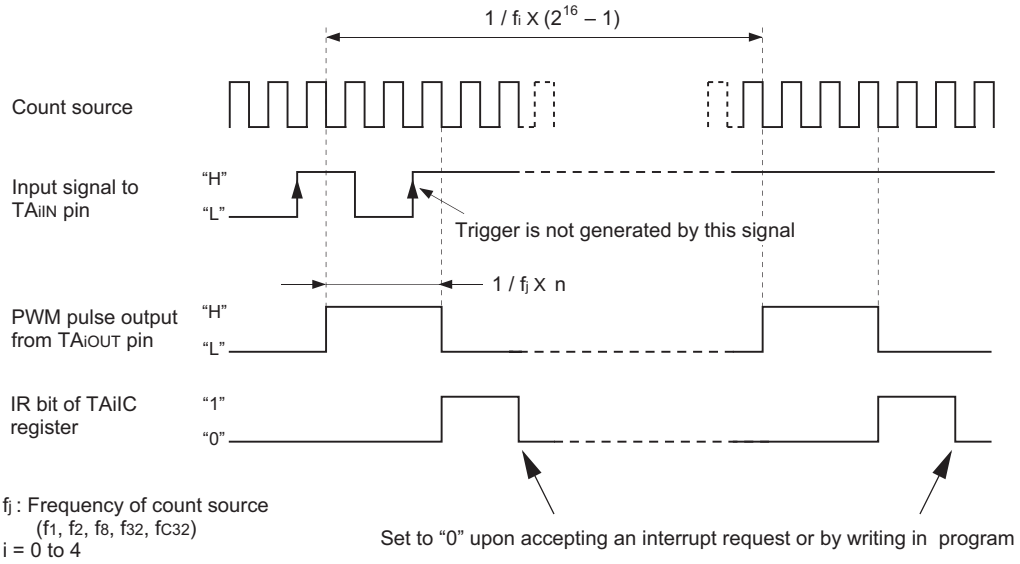
| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | 1 | 1 | 1 |

| Symbol | Address | After reset |
|---|---|---|
| TA0MR to TA4MR | 039616 to 039A16 | 0016 |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>1 1 : PWM mode (Note 1) | RW |
| TMOD1 | | | RW |
| MR0 | Must be set to “1” in PWM mode | | RW |
| MR1 | External trigger select bit (Note 2) | 0: Falling edge of input signal to TAiIN pin(Note 3)<br>1: Rising edge of input signal to TAiIN pin(Note 3) | RW |
| MR2 | Trigger select bit | 0 : Write “1” to TAiS bit in the TABSR register<br>1 : Selected by TAiTGH to TAiTGL bits | RW |
| MR3 | 16/8-bit PWM mode select bit | 0: Functions as a 16-bit pulse width modulator<br>1: Functions as an 8-bit pulse width modulator | RW |
| TCK0 | Count source select bit | b7 b6<br>0 0 : f1 or f2<br>0 1 : f8<br>1 0 : f32<br>1 1 : fC32 | RW |
| TCK1 | | | RW |

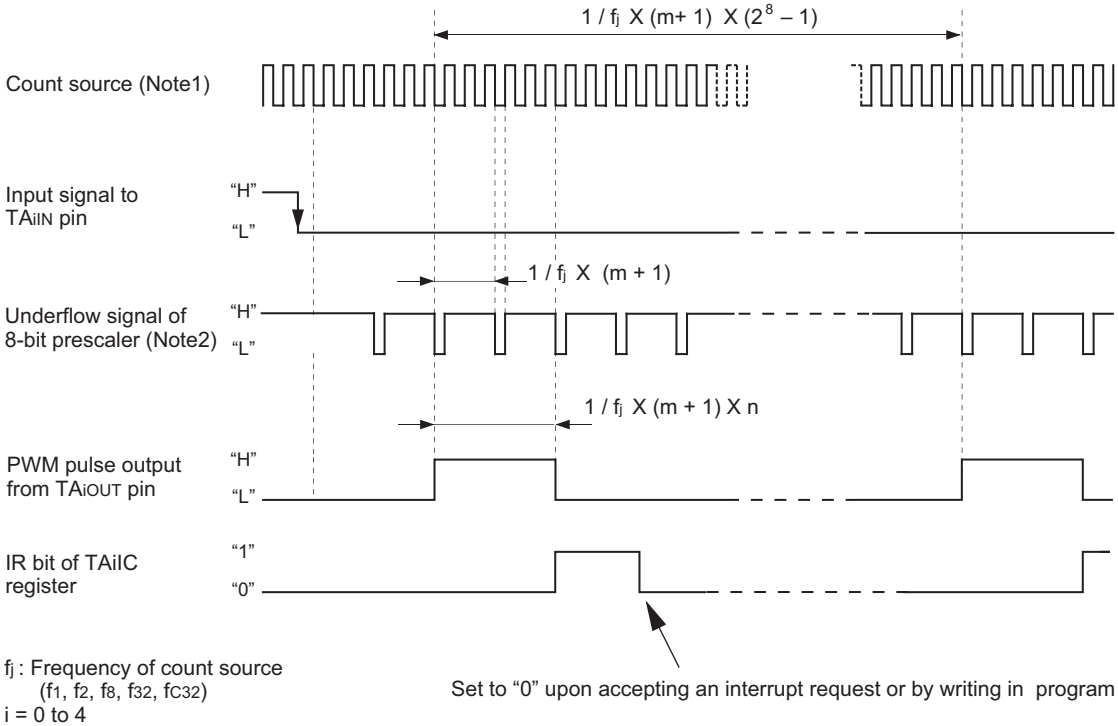
Note 1: TA0OUT pin is N-channel open drain output.
Note 2: Effective when the TAiTGH and TAiTGL bits of ONSF or TRGSR register are “00”2 (TAi IN pin input).
Note 3: The port direction bit for the TAiIN pin must be set to “0” (= input mode).

**Figure 9.11 TAiMR Register in PWM Mode**




Note 1: n = 0000$_{16}$ to FFFE$_{16}$.
Note 2: This timing diagram is for the case where the TAi register is '0003$_{16}$,' the TAiTGH and TAiTGL bits of ONSF or TRGSR register = '00$_2$' (TAiIN pin input), the MR1 bit of TAiMR register = 1 (rising edge), and the MR2 bit of TAiMR register = 1 (trigger selected by TAiTGH and TAiTGL bits).

**Figure 9.12 Example of 16-bit Pulse Width Modulator Operation**




Note 1: The 8-bit prescaler counts the count source.
Note 2: The 8-bit pulse width modulator counts the 8-bit prescaler's underflow signal.
Note 3: m = 00$_{16}$ to FF$_{16}$; n = 00$_{16}$ to FE$_{16}$.
Note 4: This timing diagram is for the case where the TAi register is '0202$_{16}$,' the TAiTGH and TAiTGL bits of ONSF or TRGSR register = '00$_2$' (TAiIN pin input), the MR1 bit of TAiMR register = 0 (falling edge), and the MR2 bit of TAiMR register = 1 (trigger selected by TAiTGH and TAiTGL bits).

**Figure 9.13 Example of 8-bit Pulse Width Modulator Operation**

## 9.2 Timer B

Figure 9.14 shows a block diagram of the timer B. Figures 9.15 and 9.16 show registers related to the timer B. Timer B supports the following three modes. Use the TMOD1 and TMOD0 bits of TBiMR register (i = 0 to 5) to select the desired mode.

- Timer mode: The timer counts an internal count source.
- Event counter mode: The timer counts pulses from an external device or overflows or underflows of other timers.
- Pulse period/pulse width measuring mode: The timer measures an external signal's pulse period or pulse width.

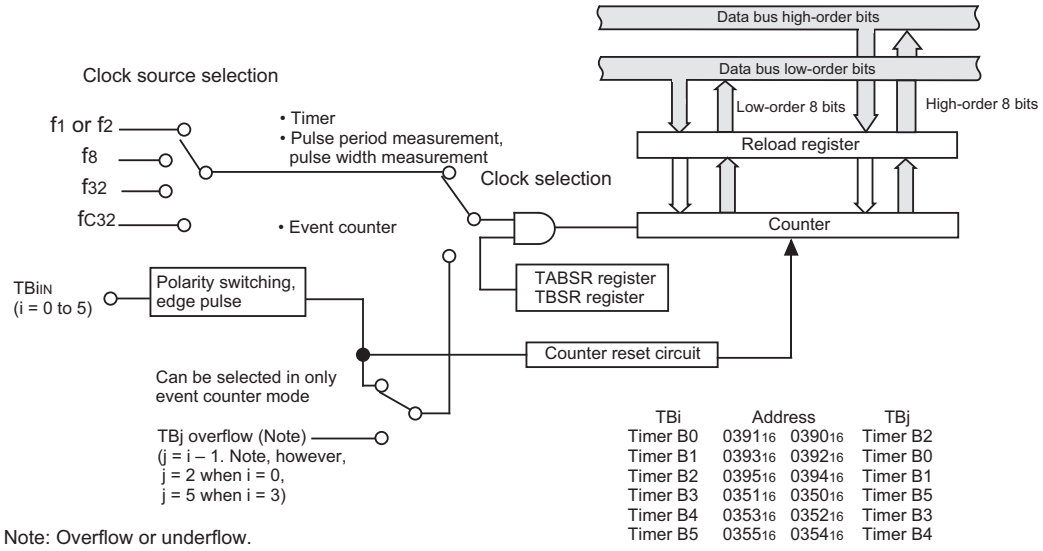



**Figure 9.14 Timer B Block Diagram**

Timer Bi mode register (i=0 to 5)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

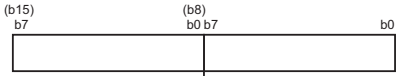
| Symbol | Address | After reset |
|---|---|---|
| TB0MR to TB2MR | 039B16 to 039D16 | 00XX00002 |
| TB3MR to TB5MR | 035B16 to 035D16 | 00XX00002 |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 0 : Timer mode<br>0 1 : Event counter mode<br>1 0 : Pulse period measurement mode, pulse width measurement mode<br>1 1 : Must not be set | RW |
| TMOD1 | | | RW |
| MR0 | ——— | Function varies with each operation mode | RW |
| MR1 | | | RW |
| MR2 | | | RW (Note 1)<br>— (Note 2) |
| MR3 | | | RO |
| TCK0 | Count source select bit | Function varies with each operation mode | RW |
| TCK1 | | | RW |

Note 1: Timer B0, timer B3.
Note 2: Timer B1, timer B2, timer B4, timer B5.

**Figure 9.15 TB0MR to TB5MR Registers**

Timer Bi register (i=0 to 5)(Note 1)

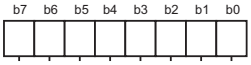



| Symbol | Address | After reset |
|---|---|---|
| TB0 | 039116, 039016 | Indeterminate |
| TB1 | 039316, 039216 | Indeterminate |
| TB2 | 039516, 039416 | Indeterminate |
| TB3 | 035116, 035016 | Indeterminate |
| TB4 | 035316, 035216 | Indeterminate |
| TB5 | 035516, 035416 | Indeterminate |

| Mode | Function | Setting range | RW |
|---|---|---|---|
| Timer mode | Divide the count source by n + 1 where n = set value | 000016 to FFFF16 | RW |
| Event counter mode | Divide the count source by n + 1 where n = set value (Note 2) | 000016 to FFFF16 | RW |
| Pulse period modulation mode, Pulse width modulation mode | Measures a pulse period or width | —— | RO |

Note 1: The register must be accessed in 16 bit units.
Note 2: The timer counts pulses from an external device or overflows or underflows of other timers.

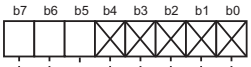
Count start flag




Symbol: TABSR
Address: 038016
After reset: 0016

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| TA0S | Timer A0 count start flag | 0 : Stops counting<br>1 : Starts counting | RW |
| TA1S | Timer A1 count start flag | | RW |
| TA2S | Timer A2 count start flag | | RW |
| TA3S | Timer A3 count start flag | | RW |
| TA4S | Timer A4 count start flag | | RW |
| TB0S | Timer B0 count start flag | | RW |
| TB1S | Timer B1 count start flag | | RW |
| TB2S | Timer B2 count start flag | | RW |

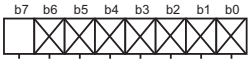
Timer B3, B4, B5 count start flag




Symbol: TBSR
Address: 034016
After reset: 000XXXXX2

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| —— (b4-b0) | Nothing is assigned. When write, set to “0”. When read, their contents are indeterminate. | | — |
| TB3S | Timer B3 count start flag | 0 : Stops counting<br>1 : Starts counting | RW |
| TB4S | Timer B4 count start flag | | RW |
| TB5S | Timer B5 count start flag | | RW |

Clock prescaler reset flag




Symbol: CPSRF
Address: 038116
After reset: 0XXXXXXX2

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| —— (b6-b0) | Nothing is assigned. When write, set to “0”. When read, their contents are indeterminate. | | — |
| CPSR | Clock prescaler reset flag | Setting this bit to “1” initializes the prescaler for the timekeeping clock. (When read, the value of this bit is “0”.) | RW |

**Figure 9.16 TB0 to TB5 Registers, TABSR Register, TBSR Register, CPSRF Register**

### 9.2.1 Timer Mode

In timer mode, the timer counts a count source generated internally (see Table 9.6). Figure 9.17 shows TBiMR register in timer mode.

**Table 9.6 Specifications in Timer Mode**

| Item | Specification |
|---|---|
| Count source | $f_1$, $f_2$, $f_8$, $f_{32}$, $f_{C32}$ |
| Count operation | • Down-count<br>• When the timer underflows, it reloads the reload register contents and continues counting |
| Divide ratio | 1/(n+1) n: set value of TBi register (i= 0 to 5) $0000_{16}$ to $FFFF_{16}$ |
| Count start condition | Set TBiS bit(Note) to "1" (= start counting) |
| Count stop condition | Set TBiS bit to "0" (= stop counting) |
| Interrupt request generation timing | Timer underflow |
| TBiIN pin function | I/O port |
| Read from timer | Count value can be read by reading TBi register |
| Write to timer | • When not counting and until the 1st count source is input after counting start<br>Value written to TBi register is written to both reload register and counter<br>• When counting (after 1st count source input)<br>Value written to TBi register is written to only reload register<br>(Transferred to counter when reloaded next) |

Note : The TB0S to TB2S bits are assigned to the TABSR register bit 5 to bit 7, and the TB3S to TB5S bits are assigned to the TBSR register bit 5 to bit 7.

Timer Bi mode register (i= 0 to 5)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | 0 | 0 |

| Symbol | Address | After reset |
|---|---|---|
| TB0MR to TB2MR | $039B_{16}$ to $039D_{16}$ | $00XX0000_2$ |
| TB3MR to TB5MR | $035B_{16}$ to $035D_{16}$ | $00XX0000_2$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 0 : Timer mode | RW |
| TMOD1 | | | RW |
| MR0 | Has no effect in timer mode<br>Can be set to "0" or "1" | | RW |
| MR1 | | | RW |
| MR2 | TB0MR, TB3MR registers<br>Must be set to "0" in timer mode | | RW |
| | TB1MR, TB2MR, TB4MR, TB5MR registers<br>Nothing is assigned. When write, set to "0". When read, its content is indeterminate | | — |
| MR3 | When write in timer mode, set to "0". When read in timer mode, its content is indeterminate. | | RO |
| TCK0 | Count source select bit | b7 b6<br>0 0 : $f_1$ or $f_2$<br>0 1 : $f_8$<br>1 0 : $f_{32}$<br>1 1 : $f_{C32}$ | RW |
| TCK1 | | | RW |

**Figure 9.17 TBiMR Register in Timer Mode**

### 9.2.2 Event Counter Mode

In event counter mode, the timer counts pulses from an external device or overflows and underflows of other timers (see Table 9.7) . Figure 9.18 shows TBiMR register in event counter mode.

**Table 9.7 Specifications in Event Counter Mode**

| Item | Specification |
|---|---|
| Count source | • External signals input to TBiIN pin (i=0 to 5) (effective edge can be selected in program)<br>• Timer Bj overflow or underflow (j=i-1, except j=2 if i=0, j=5 if i=3) |
| Count operation | • Down-count<br>• When the timer underflows, it reloads the reload register contents and continues counting |
| Divide ratio | 1/(n+1) n: set value of TBi register 0000$_{16}$ to FFFF$_{16}$ |
| Count start condition | Set TBiS bit[1] to “1” (= start counting) |
| Count stop condition | Set TBiS bit to “0” (= stop counting) |
| Interrupt request generation timing | Timer underflow |
| TBiIN pin function | Count source input |
| Read from timer | Count value can be read by reading TBi register |
| Write to timer | • When not counting and until the 1st count source is input after counting start<br>Value written to TBi register is written to both reload register and counter<br>• When counting (after 1st count source input)<br>Value written to TBi register is written to only reload register<br>(Transferred to counter when reloaded next) |

Notes:
1. The TB0S to TB2S bits are assigned to the TABSR register bit 5 to bit 7, and the TB3S to TB5S bits are assigned to the TBSR register bit 5 to bit 7.

Timer Bi mode register (i=0 to 5)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | 0 | 1 |

| Symbol | Address | After reset |
|---|---|---|
| TB0MR to TB2MR | 039B$_{16}$ to 039D$_{16}$ | 00XX0000$_2$ |
| TB3MR to TB5MR | 035B$_{16}$ to 035D$_{16}$ | 00XX0000$_2$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 1 : Event counter mode | RW |
| TMOD1 | | | RW |
| MR0 | Count polarity select bit (Note 1) | b3 b2<br>0 0 : Counts external signal's falling edges<br>0 1 : Counts external signal's rising edges<br>1 0 : Counts external signal's falling and rising edges<br>1 1 : Must not be set | RW |
| MR1 | | | RW |
| MR2 | TB0MR, TB3MR registers<br>Must be set to “0” in event count mode | | RW |
| | TB1MR, TB2MR, TB4MR, TB5MR registers<br>Nothing is assigned. When write, set to “0”. When read, its content is indeterminate. | | — |
| MR3 | When write in event counter mode, set to “0”. When read in event counter mode, its content is indeterminate. | | RO |
| TCK0 | Has no effect in event counter mode.<br>Can be set to “0” or “1”. | | RW |
| TCK1 | Event clock select | 0 : Input from TBiIN pin (Note 2)<br>1 : TBj overflow or underflow<br>(j = i – 1, except j = 2 if i = 0, j = 5 if i = 3) | RW |

Note 1: Effective when the TCK1 bit = “0” (input from TBiIN pin). If the TCK1 bit = “1” (TBj overflow or underflow), these bits can be set to “0” or “1”.
Note 2: The port direction bit for the TBiIN pin must be set to “0” (= input mode).

**Figure 9.18 TBiMR Register in Event Counter Mode**

### 9.2.3 Pulse Period and Pulse Width Measurement Mode

In pulse period and pulse width measurement mode, the timer measures pulse period or pulse width of an external signal (see Table 9.8). Figure 9.19 shows TBiMR register in pulse period and pulse width measurement mode. Figure 9.20 shows the operation timing when measuring a pulse period. Figure 9.21 shows the operation timing when measuring a pulse width.

**Table 9.8 Specifications in Pulse Period and Pulse Width Measurement Mode**

| Item | Specification |
|---|---|
| Count source | $f_1$, $f_2$, $f_8$, $f_{32}$, $f_{C32}$ |
| Count operation | • Up-count<br>• Counter value is transferred to reload register at an effective edge of measurement pulse. The counter value is set to "$0000_{16}$" to continue counting. |
| Count start condition | Set TBiS (i=0 to 5) bit[3] to "1" (= start counting) |
| Count stop condition | Set TBiS bit to "0" (= stop counting) |
| Interrupt request generation timing | • When an effective edge of measurement pulse is input[1]<br>• Timer overflow. When an overflow occurs, MR3 bit of TBiMR register is set to "1" (overflowed) simultaneously. MR3 bit is cleared to "0" (no overflow) by writing to TBiMR register at the next count timing or later after MR3 bit was set to "1". At this time, make sure TBiS bit is set to "1" (start counting). |
| $TBi_{IN}$ pin function | Measurement pulse input |
| Read from timer | Contents of the reload register (measurement result) can be read by reading TBi register[2] |
| Write to timer | Value written to TBi register is written to neither reload register nor counter |

Notes:
1. Interrupt request is not generated when the first effective edge is input after the timer started counting.
2. Value read from TBi register is indeterminate until the second valid edge is input after the timer starts counting.
3. The TB0S to TB2S bits are assigned to the TABSR register bit 5 to bit 7, and the TB3S to TB5S bits are assigned to the TBSR register bit 5 to bit 7.

Timer Bi mode register (i=0 to 5)

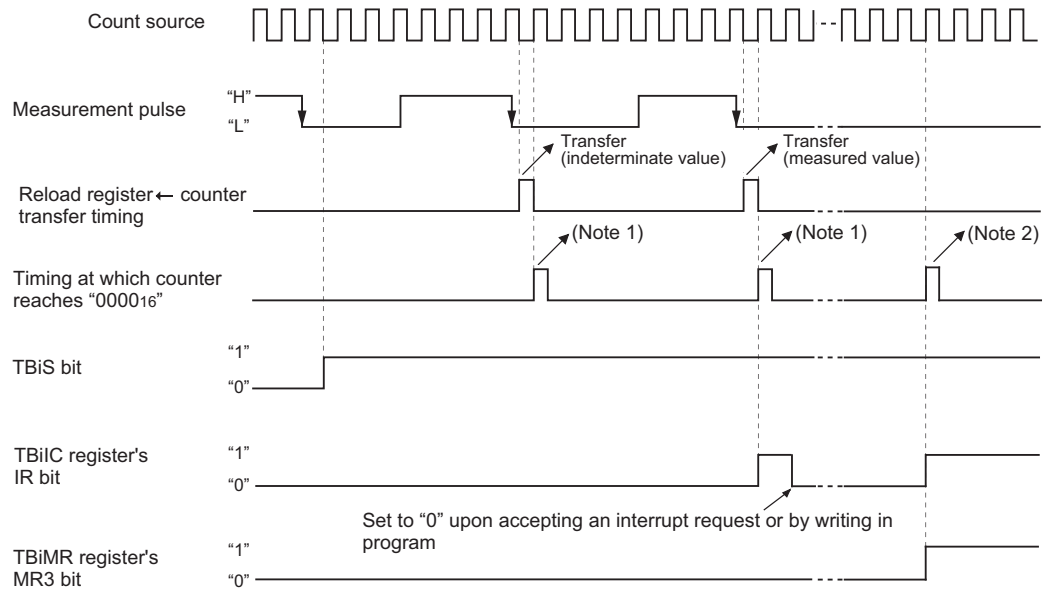
| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | 1 | 0 |

| Symbol | Address | After reset |
|---|---|---|
| TB0MR to TB2MR | $039B_{16}$ to $039D_{16}$ | $00XX0000_2$ |
| TB3MR to TB5MR | $035B_{16}$ to $035D_{16}$ | $00XX0000_2$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>1 0 : Pulse period / pulse width measurement mode | RW |
| TMOD1 | | | RW |
| MR0 | Measurement mode select bit | b3 b2<br>0 0 : Pulse period measurement (Measurement between a falling edge and the next falling edge of measured pulse)<br>0 1 : Pulse period measurement (Measurement between a rising edge and the next rising edge of measured pulse)<br>1 0 : Pulse width measurement (Measurement between a falling edge and the next rising edge of measured pulse and between a rising edge and the next falling edge)<br>1 1 : Must not be set. | RW |
| MR1 | | | RW |
| MR2 | TB0MR and TB3MR registers<br>Must be set to "0" in pulse period and pulse width measurement mode | | RW |
| | TB1MR, TB2MR, TB4MR, TB5MR registers<br>Nothing is assigned. When write, set to "0". When read, its content turns out to be indeterminate. | | — |
| MR3 | Timer Bi overflow flag ( Note) | 0 : Timer did not overflow<br>1 : Timer has overflowed | RO |
| TCK0 | Count source select bit | b7 b6<br>0 0 : $f_1$ or $f_2$<br>0 1 : $f_8$<br>1 0 : $f_{32}$<br>1 1 : $f_{C32}$ | RW |
| TCK1 | | | RW |

Note: This flag is indeterminate after reset. When the TBiS bit = 1 (start counting), the MR3 bit is cleared to "0" (no overflow) by writing to the TBiMR register at the next count timing or later after the MR3 bit was set to "1" (overflowed). The MR3 bit cannot be set to "1" in a program. The TB0S to TB2S bits are assigned to the TABSR register's bit 5 to bit 7, and the TB3S to TB5S bits are assigned to the TBSR register's bit 5 to bit 7.

**Figure 9.19 TBiMR Register in Pulse Period and Pulse Width Measurement Mode**

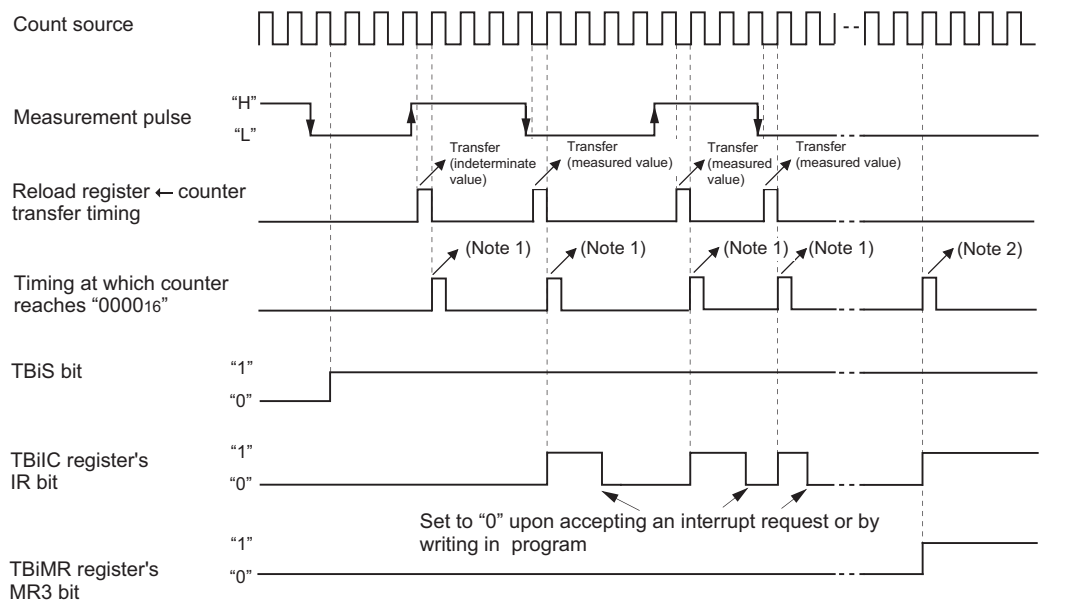



The TB0S to TB2S bits are assigned to the TABSR register's bit 5 to bit 7, and the TB3S to TB5S bits are assigned to the TBSR register's bit 5 to bit 7.

i = 0 to 5

Note 1: Counter is initialized at completion of measurement.
Note 2: Timer has overflowed.
Note 3: This timing diagram is for the case where the TBiMR register's MR1 to MR0 bits are "00₂" (measure the interval from falling edge to falling edge of the measurement pulse).

**Figure 9.20 Operation timing when measuring a pulse period**




The TB0S to TB2S bits are assigned to the TABSR register's bit 5 to bit 7, and the TB3S to TB5S bits are assigned to the TBSR register's bit 5 to bit 7.

i = 0 to 5

Note 1: Counter is initialized at completion of measurement.
Note 2: Timer has overflowed.
Note 3: This timing diagram is for the case where the TBiMR register's MR1 to MR0 bits are "10₂" (measure the interval from a falling edge to the next rising edge and the interval from a rising edge to the next falling edge of the measurement pulse).

**Figure 9.21 Operation timing when measuring a pulse width**

# 10. Serial I/O

Serial I/O is configured with five channels: UART0 to UART2, SI/O3 and SI/O4.

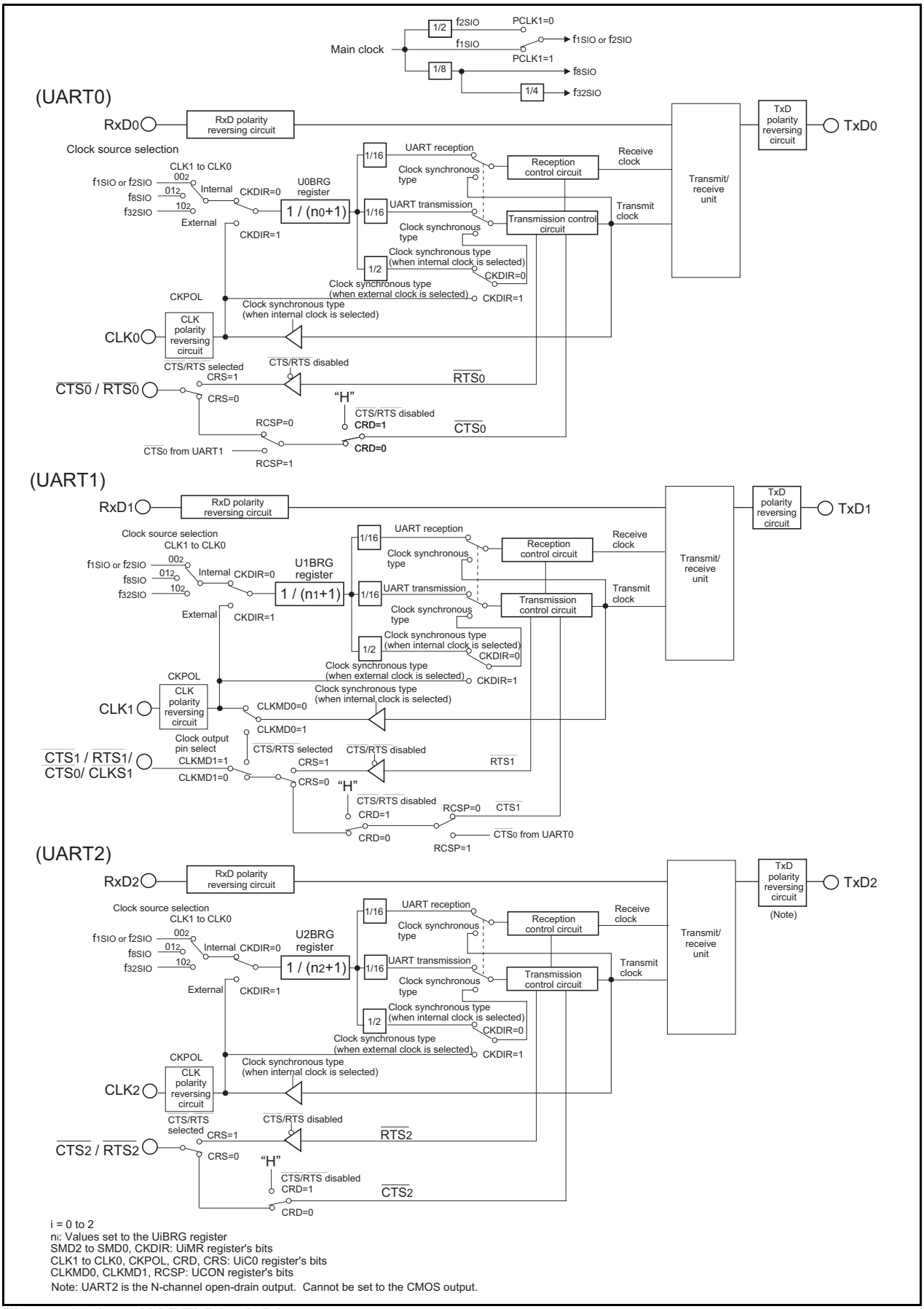
## 10.1 UARTi (i=0 to 2)

UARTi each have an exclusive timer to generate a transfer clock, so they operate independently of each other. Figure 10.1 shows the block diagram of UARTi. Figures 10.2 shows the block diagram of the UARTi transmit/ receive.

UARTi has the following modes:

- Clock synchronous serial I/O mode
- Clock asynchronous serial I/O mode (UART mode).
- Special mode 1 ($I^2C$ mode)
- Special mode 2
- Special mode 3 (Bus collision detection function, IE mode) : UART0, UART1
- Special mode 4 (SIM mode) : UART2

Figures 10.3 to 10.8 show the UARTi-related registers.
Refer to tables listing each mode for register setting.




i = 0 to 2
ni: Values set to the UiBRG register
SMD2 to SMD0, CKDIR: UiMR register's bits
CLK1 to CLK0, CKPOL, CRD, CRS: UiC0 register's bits
CLKMD0, CLKMD1, RCSP: UCON register's bits
Note: UART2 is the N-channel open-drain output. Cannot be set to the CMOS output.

**Figure 10.1 UARTi Block Diagram**

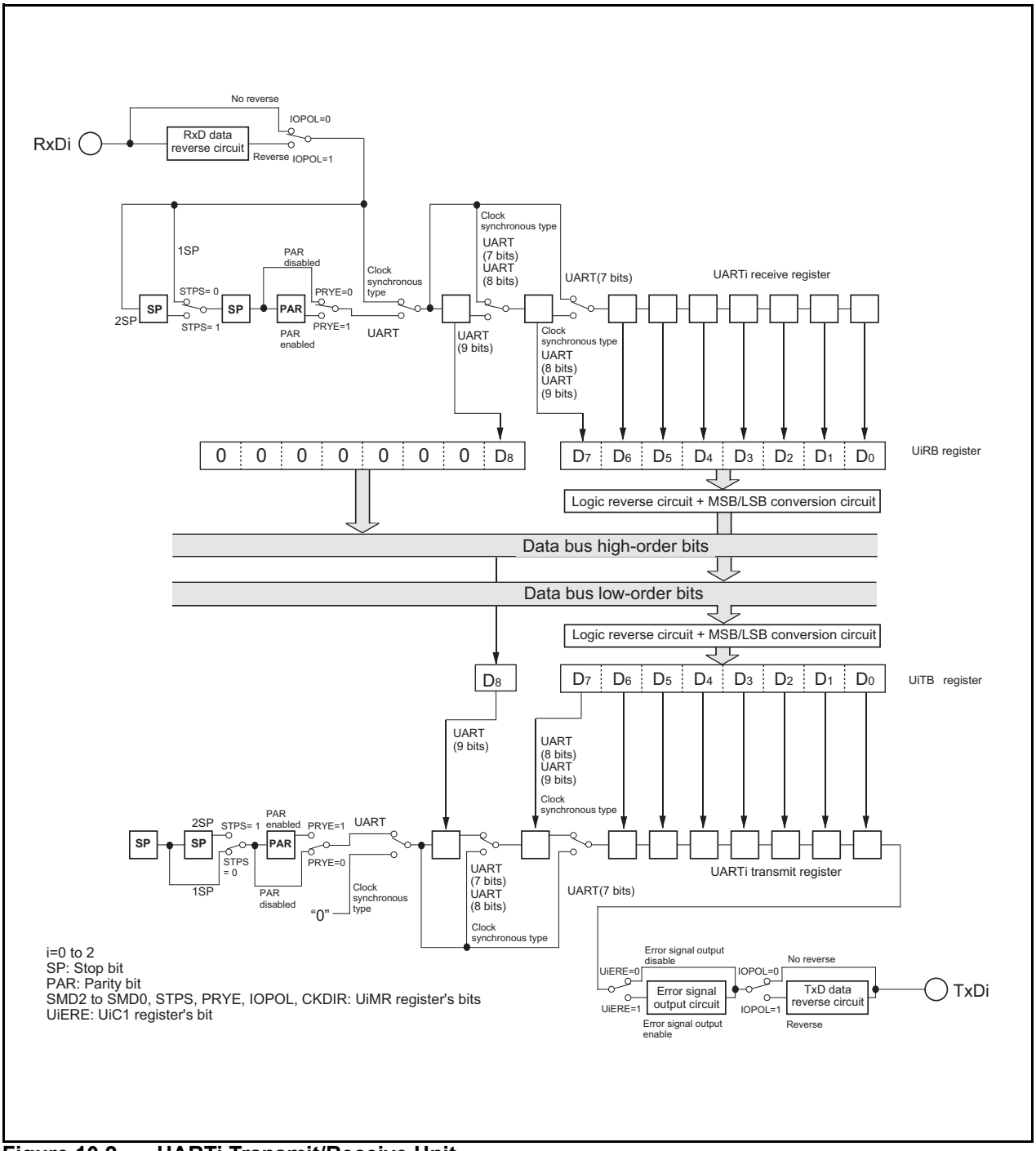
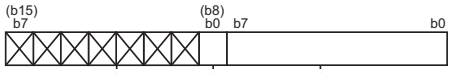



**Figure 10.2 UARTi Transmit/Receive Unit**

UARTi transmit buffer register (i=0 to 2)(Note)

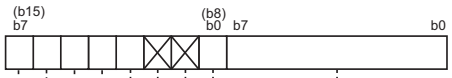



| Symbol | Address | After reset |
|---|---|---|
| U0TB | $03A3_{16}$-$03A2_{16}$ | Indeterminate |
| U1TB | $03AB_{16}$-$03AA_{16}$ | Indeterminate |
| U2TB | $037B_{16}$-$037A_{16}$ | Indeterminate |

| Function | RW |
|---|---|
| Transmit data | WO |
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminate. | — |

Note: Use MOV instruction to write to this register.

UARTi receive buffer register (i=0 to 2)

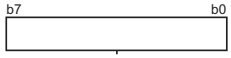



| Symbol | Address | After reset |
|---|---|---|
| U0RB | $03A7_{16}$-$03A6_{16}$ | Indeterminate |
| U1RB | $03AF_{16}$-$03AE_{16}$ | Indeterminate |
| U2RB | $037F_{16}$-$037E_{16}$ | Indeterminate |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| — (b7-b0) | ——— | Receive data ($D_7$ to $D_0$) | RO |
| — (b8) | ——— | Receive data ($D_8$) | RO |
| — (b10-b9) | Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”. | | — |
| ABT | Arbitration lost detecting flag (Note 2) | 0 : Not detected<br>1 : Detected | RW |
| OER | Overrun error flag (Note 1) | 0 : No overrun error<br>1 : Overrun error found | RO |
| FER | Framing error flag (Note 1) | 0 : No framing error<br>1 : Framing error found | RO |
| PER | Parity error flag (Note 1) | 0 : No parity error<br>1 : Parity error found | RO |
| SUM | Error sum flag (Note 1) | 0 : No error<br>1 : Error found | RO |

Note 1: When the UiMR register’s SMD2 to SMD0 bits = “$000_2$” (serial I/O disabled) or the UiC1 register’s RE bit = “0” (reception disabled), all of the SUM, PER, FER and OER bits are set to “0” (no error). The SUM bit is set to “0” (no error) when all of the PER, FER and OER bits = “0” (no error). Also, the PER and FER bits are set to “0” by reading the lower byte of the UiRB register.
Note 2: The ABT bit is set to “0” by writing “0” in a program. (Writing “1” has no effect.)

UARTi bit rate generator (i=0 to 2)(Notes 1, 2)



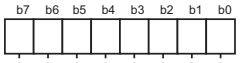

| Symbol | Address | After reset |
|---|---|---|
| U0BRG | $03A1_{16}$ | Indeterminate |
| U1BRG | $03A9_{16}$ | Indeterminate |
| U2BRG | $0379_{16}$ | Indeterminate |

| Function | Setting range | RW |
|---|---|---|
| Assuming that set value = n, UiBRG divides the count source by n + 1 | $00_{16}$ to $FF_{16}$ | WO |

Note 1: Write to this register while serial I/O is neither transmitting nor receiving.
Note 2: Use MOV instruction to write to this register.

**Figure 10.3 U0TB to U2TB Register, U0RB to U2RB Register, and U0BRG to U2BRG Register**

UARTi transmit/receive mode register (i=0 to 2)

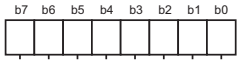



Symbol U0MR to U2MR | Address 03A0₁₆, 03A8₁₆, 037816 | After reset 00₁₆

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| SMD0 | Serial I/O mode select bit (Note 2) | b2 b1 b0<br>0 0 0 : Serial I/O disabled<br>0 0 1 : Clock synchronous serial I/O mode<br>0 1 0 : I²C mode (Note 3)<br>1 0 0 : UART mode transfer data 7 bits long<br>1 0 1 : UART mode transfer data 8 bits long<br>1 1 0 : UART mode transfer data 9 bits long<br>Must not be set except above | RW |
| SMD1 | | | RW |
| SMD2 | | | RW |
| CKDIR | Internal/external clock select bit | 0 : Internal clock<br>1 : External clock (Note 1) | RW |
| STPS | Stop bit length select bit | 0 : One stop bit<br>1 : Two stop bits | RW |
| PRY | Odd/even parity select bit | Effective when PRYE = 1<br>0 : Odd parity<br>1 : Even parity | RW |
| PRYE | Parity enable bit | 0 : Parity disabled<br>1 : Parity enabled | RW |
| IOPOL | TxD, RxD I/O polarity reverse bit | 0 : No reverse<br>1 : Reverse | RW |

Note 1: Set the corresponding port direction bit for each CLKi pin to "0" (input mode).
Note 2: To receive data, set the corresponding port direction bit for each RxDi pin to "0" (input mode).
Note 3: Set the corresponding port direction bit for SCL and SDA pins to "0" (input mode).

UARTi transmit/receive control register 0 (i=0 to 2)



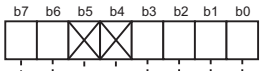

Symbol U0C0 to U2C0 | Address 03A4₁₆, 03AC₁₆, 037C₁₆ | After reset 00001000₂

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| CLK0 | BRG count source select bit | b1 b0<br>0 0 : f1SIO or f2SIO is selected<br>0 1 : f8SIO is selected<br>1 0 : f32SIO is selected<br>1 1 : Must not be set | RW |
| CLK1 | | | RW |
| CRS | CTS/RTS function select bit (Note 4) | Effective when CRD = 0<br>0 : CTS function is selected (Note 1)<br>1 : RTS function is selected | RW |
| TXEPT | Transmit register empty flag | 0 : Data present in transmit register (during transmission)<br>1 : No data present in transmit register (transmission completed) | RO |
| CRD | CTS/RTS disable bit | 0 : CTS/RTS function enabled<br>1 : CTS/RTS function disabled<br>(P6₀, P6₄ and P7₃ can be used as I/O ports) | RW |
| NCH | Data output select bit (Note 2) | 0 : TxDi/SDAi and SCLi pins are CMOS output<br>1 : TxDi/SDAi and SCLi pins are N-channel open-drain output | RW |
| CKPOL | CLK polarity select bit | 0 : Transmit data is output at falling edge of transfer clock and receive data is input at rising edge<br>1 : Transmit data is output at rising edge of transfer clock and receive data is input at falling edge | RW |
| UFORM | Transfer format select bit (Note 3) | 0 : LSB first<br>1 : MSB first | RW |

Note 1: Set the corresponding port direction bit for each CTSi pin to "0" (input mode).
Note 2: TxD2/SDA2 and SCL2 are N-channel open-drain output. Cannot be set to the CMOS output. Set the NCH bit of the U2C0 register to "0".
Note 3: Effective for clock synchronous serial I/O mode and UART mode transfer data 8 bits long.
Note 4: CTS1/RTS1 can be used when the UCON register's CLKMD1 bit = "0" (only CLK1 output) and the UCON register's RCSP bit = "0" (CTS0/RTS0 not separated).

**Figure 10.4 U0MR to U2MR Register and U0C0 to U2C0 Register**

UARTi transmit/receive control register 1 (i=0, 1)




Symbol: U0C1, U1C1 | Address: 03A5$_{16}$,03AD$_{16}$ | After reset: 00000010$_2$

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| TE | Transmit enable bit | 0 : Transmission disabled<br>1 : Transmission enabled | RW |
| TI | Transmit buffer empty flag | 0 : Data present in UiTB register<br>1 : No data present in UiTB register | RO |
| RE | Receive enable bit | 0 : Reception disabled<br>1 : Reception enabled | RW |
| RI | Receive complete flag | 0 : No data present in UiRB register<br>1 : Data present in UiRB register | RO |
| — (b5-b4) | Nothing is assigned.<br>When write, set “0”. When read, these contents are “0”. | | — |
| UiLCH | Data logic select bit | 0 : No reverse<br>1 : Reverse | RW |
| UiERE | Error signal output enable bit | 0 : Output disabled<br>1 : Output enabled | RW |

UART2 transmit/receive control register 1




Symbol: U2C1 | Address: 037D$_{16}$ | After reset: 00000010$_2$

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| TE | Transmit enable bit | 0 : Transmission disabled<br>1 : Transmission enabled | RW |
| TI | Transmit buffer empty flag | 0 : Data present in U2TB register<br>1 : No data present in U2TB register | RO |
| RE | Receive enable bit | 0 : Reception disabled<br>1 : Reception enabled | RW |
| RI | Receive complete flag | 0 : No data present in U2RB register<br>1 : Data present in U2RB register | RO |
| U2IRS | UART2 transmit interrupt cause select bit | 0 : Transmit buffer empty (TI = 1)<br>1 : Transmit is completed (TXEPT = 1) | RW |
| U2RRM | UART2 continuous receive mode enable bit | 0 : Continuous receive mode disabled<br>1 : Continuous receive mode enabled | RW |
| U2LCH | Data logic select bit | 0 : No reverse<br>1 : Reverse | RW |
| U2ERE | Error signal output enable bit | 0 : Output disabled<br>1 : Output enabled | RW |

**Figure 10.5 U0C1 to U2C1 Registers**

UART transmit/receive control register 2

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| X | | | | | | | |

Symbol: UCON | Address: 03B0$_{16}$ | After reset: X0000000$_2$

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| U0IRS | UART0 transmit interrupt cause select bit | 0 : Transmit buffer empty (TI = 1)<br>1 : Transmission completed (TXEPT = 1) | RW |
| U1IRS | UART1 transmit interrupt cause select bit | 0 : Transmit buffer empty (TI = 1)<br>1 : Transmission completed (TXEPT = 1) | RW |
| U0RRM | UART0 continuous receive mode enable bit | 0 : Continuous receive mode disabled<br>1 : Continuous receive mode enable | RW |
| U1RRM | UART1 continuous receive mode enable bit | 0 : Continuous receive mode disabled<br>1 : Continuous receive mode enabled | RW |
| CLKMD0 | UART1 CLK/CLKS select bit 0 | Effective when CLKMD1 = "1"<br>0 : Clock output from CLK1<br>1 : Clock output from CLKS1 | RW |
| CLKMD1 | UART1 CLK/CLKS select bit 1 (Note) | 0 : CLK output is only CLK1<br>1 : Transfer clock output from multiple pins function selected | RW |
| RCSP | Separate UART0 $\overline{CTS}/\overline{RTS}$ bit | 0 : $\overline{CTS}/\overline{RTS}$ shared pin<br>1 : $\overline{CTS}/\overline{RTS}$ separated ($\overline{CTS}_0$ supplied from the P6$_4$ pin) | RW |
| — (b7) | Nothing is assigned. When write, set "0". When read, its content is indeterminate. | | — |

Note: When using multiple transfer clock output pins, make sure the following conditions are met:
U1MR register's CKDIR bit = "0" (internal clock)

UART2 special mode register (i=0 to 2)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| X | | | | 0 | | | |

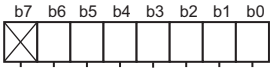
Symbol: U0SMR to U2SMR | Address: 036F$_{16}$, 0373$_{16}$, 0377$_{16}$ | After reset: X0000000$_2$

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| IICM | I$^2$C mode select bit | 0 : Other than I$^2$C mode<br>1 : I$^2$C mode | RW |
| ABC | Arbitration lost detecting flag control bit | 0 : Update per bit<br>1 : Update per byte | RW |
| BBS | Bus busy flag | 0 : STOP condition detected<br>1 : START condition detected (busy) | RW (Note1) |
| — (b3) | Reserved bit | Set to "0" | RW |
| ABSCS | Bus collision detect sampling clock select bit | 0 : Rising edge of transfer clock<br>1 : Underflow signal of timer Aj (Note 2) | RW |
| ACSE | Auto clear function select bit of transmit enable bit | 0 : No auto clear function<br>1 : Auto clear at occurrence of bus collision | RW |
| SSS | Transmit start condition select bit | 0 : Not synchronized to RxDi<br>1 : Synchronized to RxDi (Note 3) | RW |
| — (b7) | Nothing is assigned. When write, set "0". When read, its content is indeterminate. | | — |

Note 1: The BBS bit is set to "0" by writing "0" in a program. (Writing "1" has no effect.).
Note 2: Underflow signal of timer A3 in UART0, underflow signal of timer A4 in UART1, underflow signal of timer A0 in UART2.
Note 3: When a transfer begins, the SSS bit is set to "0" (Not synchronized to RxDi)

**Figure 10.6 UCON Register and U0SMR to U2SMR Registers**

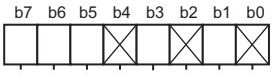
UARTi special mode register 2 (i=0 to 2)




| Symbol | Address | After reset |
|---|---|---|
| U0SMR2 to U2SMR2 | $036E_{16}$, $0372_{16}$, $0376_{16}$ | $X0000000_2$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| IICM2 | $I^2C$ mode select bit 2 | Refer to Table 10.12 | RW |
| CSC | Clock-synchronous bit | 0 : Disabled<br>1 : Enabled | RW |
| SWC | SCL wait output bit | 0 : Disabled<br>1 : Enabled | RW |
| ALS | SDA output stop bit | 0 : Disabled<br>1 : Enabled | RW |
| STAC | UARTi initialization bit | 0 : Disabled<br>1 : Enabled | RW |
| SWC2 | SCL wait output bit 2 | 0: Transfer clock<br>1: "L" output | RW |
| SDHI | SDA output disable bit | 0: Enabled<br>1: Disabled (high impedance) | RW |
| — (b7) | Nothing is assigned. When write, set "0". When read, its content is indeterminate. | | — |

UARTi special mode register 3 (i=0 to 2)




| Symbol | Address | After reset |
|---|---|---|
| U0SMR3 to U2SMR3 | $036D_{16}$, $0371_{16}$, $0375_{16}$ | $000X0X0X_2$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| — (b0) | Nothing is assigned.<br>When write, set "0". When read, its content is indeterminate. | | — |
| CKPH | Clock phase set bit | 0 : Without clock delay<br>1 : With clock delay | RW |
| — (b2) | Nothing is assigned.<br>When write, set "0". When read, its content is indeterminate. | | — |
| NODC | Clock output select bit | 0 : CLKi is CMOS output<br>1 : CLKi is N-channel open drain output | RW |
| — (b4) | Nothing is assigned.<br>When write, set "0". When read, its content is indeterminate. | | — |
| DL0 | SDAi digital delay setup bit (Note 1, Note 2) | b7 b6 b5<br>0 0 0 : Without delay<br>0 0 1 : 1 to 2 cycle(s) of UiBRG count source<br>0 1 0 : 2 to 3 cycles of UiBRG count source<br>0 1 1 : 3 to 4 cycles of UiBRG count source<br>1 0 0 : 4 to 5 cycles of UiBRG count source<br>1 0 1 : 5 to 6 cycles of UiBRG count source<br>1 1 0 : 6 to 7 cycles of UiBRG count source<br>1 1 1 : 7 to 8 cycles of UiBRG count source | RW |
| DL1 | | | RW |
| DL2 | | | RW |

Note 1 : The DL2 to DL0 bits are used to generate a delay in SDAi output by digital means during $I^2C$ mode. In other than $I^2C$ mode, set these bits to "$000_2$" (no delay).
Note 2 : The amount of delay varies with the load on SCLi and SDAi pins. Also, when using an external clock, the amount of delay increases by about 100 ns.

**Figure 10.7 U0SMR2 to U2SMR2 Registers and U0SMR3 to U2SMR3 Registers**

UARTi special mode register 4 (i=0 to 2)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

| Symbol | Address | After reset |
|---|---|---|
| U0SMR4 to U2SMR4 | $036C_{16}$, $0370_{16}$, $0374_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| STAREQ | Start condition generate bit (Note) | 0 : Clear<br>1 : Start | RW |
| RSTAREQ | Restart condition generate bit (Note) | 0 : Clear<br>1 : Start | RW |
| STPREQ | Stop condition generate bit (Note) | 0 : Clear<br>1 : Start | RW |
| STSPSEL | SCL,SDA output select bit | 0 : Start and stop conditions not output<br>1 : Start and stop conditions output | RW |
| ACKD | ACK data bit | 0 : ACK<br>1 : NACK | RW |
| ACKC | ACK data output enable bit | 0 : Serial I/O data output<br>1 : ACK data output | RW |
| SCLHI | SCL output stop enable bit | 0 : Disabled<br>1 : Enabled | RW |
| SWC9 | SCL wait bit 3 | 0 : SCL “L” hold disabled<br>1 : SCL “L” hold enabled | RW |

Note: Set to “0” when each condition is generated.

**Figure 10.8 U0SMR4 to U2SMR4 Registers**

## 10.2 Clock Synchronous serial I/O Mode

The clock synchronous serial I/O mode uses a transfer clock to transmit and receive data. Table 10.1 lists the specifications of the clock synchronous serial I/O mode. Table 10.2 lists the registers used in clock synchronous serial I/O mode and the register values set.

**Table 10.1 Clock Synchronous Serial I/O Mode Specifications**

| Item | Specification |
|---|---|
| Transfer data format | • Transfer data length: 8 bits |
| Transfer clock | • UiMR(i=0 to 2) register's CKDIR bit = "0" (internal clock) : fj/ 2(n+1)<br>fj = $f_{1SIO}$, $f_{2SIO}$, $f_{8SIO}$, $f_{32SIO}$. n: Setting value of UiBRG register $00_{16}$ to $FF_{16}$<br>• CKDIR bit = "1" (external clock) : Input from CLKi pin |
| Transmission, reception control | • Selectable from $\overline{CTS}$ function, $\overline{RTS}$ function or $\overline{CTS}/\overline{RTS}$ function disable |
| Transmission start condition | • Before transmission can start, the following requirements must be met (Note 1)<br>– The TE bit of UiC1 register= 1 (transmission enabled)<br>– The TI bit of UiC1 register = 0 (data present in UiTB register)<br>– If $\overline{CTS}$ function is selected, input on the $\overline{CTS}$i pin = "L" |
| Reception start condition | • Before reception can start, the following requirements must be met (Note 1)<br>– The RE bit of UiC1 register= 1 (reception enabled)<br>– The TE bit of UiC1 register= 1 (transmission enabled)<br>– The TI bit of UiC1 register= 0 (data present in the UiTB register) |
| Interrupt request generation timing | • For transmission, one of the following conditions can be selected<br>– The UiIRS bit (Note 3) = 0 (transmit buffer empty): when transferring data from the UiTB register to the UARTi transmit register (at start of transmission)<br>– The UiIRS bit =1 (transfer completed): when the serial I/O finished sending data from the UARTi transmit register<br>• For reception<br>When transferring data from the UARTi receive register to the UiRB register (at completion of reception) |
| Error detection | • Overrun error (Note 2)<br>This error occurs if the serial I/O started receiving the next data before reading the UiRB register and received the 7th bit of the next data |
| Select function | • CLK polarity selection<br>Transfer data input/output can be chosen to occur synchronously with the rising or the falling edge of the transfer clock<br>• LSB first, MSB first selection<br>Whether to start sending/receiving data beginning with bit 0 or beginning with bit 7 can be selected<br>• Continuous receive mode selection<br>Reception is enabled immediately by reading the UiRB register<br>• Switching serial data logic<br>This function reverses the logic value of the transmit/receive data<br>• Transfer clock output from multiple pins selection (UART1)<br>The output pin can be selected in a program from two UART1 transfer clock pins that have been set<br>• Separate $\overline{CTS}/\overline{RTS}$ pins (UART0)<br>$\overline{CTS_0}$ and $\overline{RTS_0}$ are input/output from separate pins |

Note 1: When an external clock is selected, the conditions must be met while if the UiC0 register's CKPOL bit = "0" (transmit data output at the falling edge and the receive data taken in at the rising edge of the transfer clock), the external clock is in the high state; if the UiC0 register's CKPOL bit = "1" (transmit data output at the rising edge and the receive data taken in at the falling edge of the transfer clock), the external clock is in the low state.
Note 2: If an overrun error occurs, the value of UiRB register will be indeterminate. The IR bit of SiRIC register does not change.
Note 3: The U0IRS and U1IRS bits respectively are the UCON register bits 0 and 1; the U2IRS bit is the U2C1 register bit 4.

**Table 10.2 Registers to Be Used and Settings in Clock Synchronous Serial I/O Mode**

| Register | Bit | Function |
|---|---|---|
| UiTB(Note3) | 0 to 7 | Set transmission data |
| UiRB(Note3) | 0 to 7 | Reception data can be read |
| | OER | Overrun error flag |
| UiBRG | 0 to 7 | Set a transfer rate |
| UiMR(Note3) | SMD2 to SMD0 | Set to “001$_2$” |
| | CKDIR | Select the internal clock or external clock |
| | IOPOL | Set to “0” |
| UiC0 | CLK1 to CLK0 | Select the count source for the UiBRG register |
| | CRS | Select $\overline{CTS}$ or $\overline{RTS}$ to use |
| | TXEPT | Transmit register empty flag |
| | CRD | Enable or disable the $\overline{CTS}$ or $\overline{RTS}$ function |
| | NCH | Select TxDi pin output mode (Note 2) |
| | CKPOL | Select the transfer clock polarity |
| | UFORM | Select the LSB first or MSB first |
| UiC1 | TE | Set this bit to “1” to enable transmission/reception |
| | TI | Transmit buffer empty flag |
| | RE | Set this bit to “1” to enable reception |
| | RI | Reception complete flag |
| | U2IRS (Note 1) | Select the source of UART2 transmit interrupt |
| | U2RRM (Note 1) | Set this bit to “1” to use continuous receive mode |
| | UiLCH | Set this bit to “1” to use inverted data logic |
| | UiERE | Set to “0” |
| UiSMR | 0 to 7 | Set to “0” |
| UiSMR2 | 0 to 7 | Set to “0” |
| UiSMR3 | 0 to 2 | Set to “0” |
| | NODC | Select clock output mode |
| | 4 to 7 | Set to “0” |
| UiSMR4 | 0 to 7 | Set to “0” |
| UCON | U0IRS, U1IRS | Select the source of UART0/UART1 transmit interrupt |
| | U0RRM, U1RRM | Set this bit to “1” to use continuous receive mode |
| | CLKMD0 | Select the transfer clock output pin when CLKMD1 = 1 |
| | CLKMD1 | Set this bit to “1” to output UART1 transfer clock from two pins |
| | RCSP | Set this bit to “1” to accept as input the UART0 $\overline{CTS_0}$ signal from the P6$_4$ pin |
| | 7 | Set to “0” |

Note 1: Set the U0C1 and U1C1 register bit 4 and bit 5 to “0”. The U0IRS, U1IRS, U0RRM and U1RRM bits are in the UCON register.

Note 2: TxD2 pin is N channel open-drain output. Set the U2C0 register's NCH bit to “0”.

Note 3: Not all register bits are described above. Set those bits to “0” when writing to the registers in clock synchronous serial I/O mode.

i=0 to 2

Table 10.3 lists the functions of the input/output pins during clock synchronous serial I/O mode. Table 10.3 shows pin functions for the case where the multiple transfer clock output pin select function is deselected. Table 10.4 lists the P64 pin functions during clock synchronous serial I/O mode. Note that for a period from when the UARTi operation mode is selected to when transfer starts, the TxDi pin outputs an "H". (If the N-channel open-drain output is selected, this pin is in a high-impedance state.)

**Table 10.3 Pin Functions (When Not Select Multiple Transfer Clock Output Pin Function)**

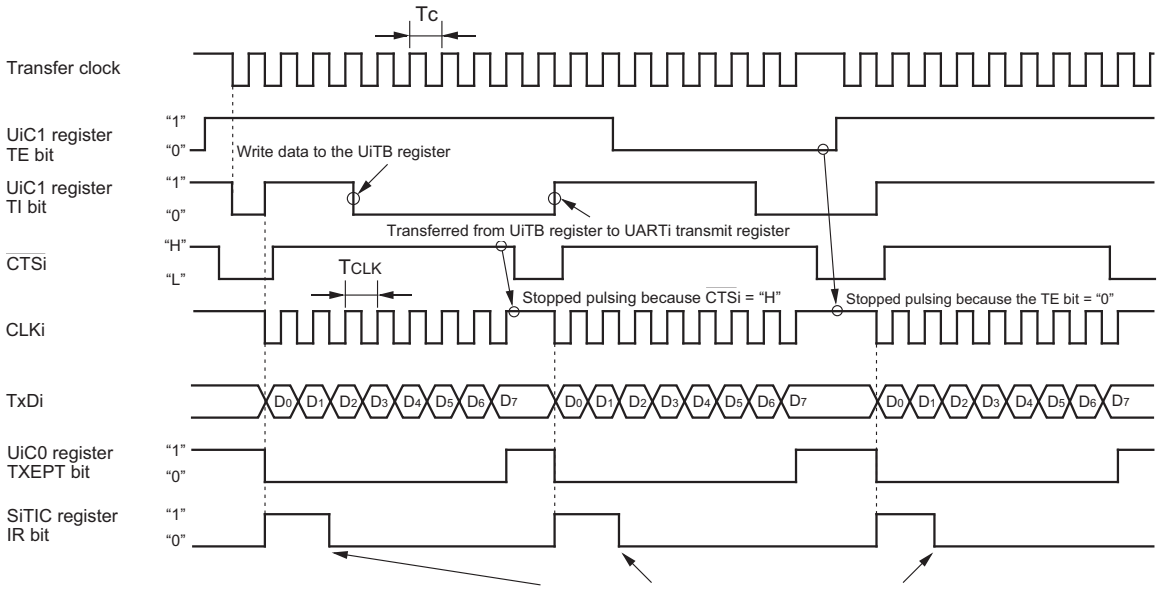
| Pin name | Function | Method of selection |
|---|---|---|
| TxDi (i = 0 to 2) (P63, P67, P70) | Serial data output | (Outputs dummy data when performing reception only) |
| RxDi (P62, P66, P71) | Serial data input | PD6 register's PD6_2 bit=0, PD6_6 bit=0, PD7 register's PD7_1 bit=0<br>(Can be used as an input port when performing transmission only) |
| CLKi (P61, P65, P72) | Transfer clock output | UiMR register's CKDIR bit=0 |
| | Transfer clock input | UiMR register's CKDIR bit=1<br>PD6 register's PD6_1 bit=0, PD6_5 bit=0, PD7 register's PD7_2 bit=0 |
| $\overline{CTS}$i/$\overline{RTS}$i (P60, P64, P73) | $\overline{CTS}$ input | UiC0 register's CRD bit=0<br>UiC0 register's CRS bit=0<br>PD6 register's PD6_0 bit=0, PD6_4 bit=0, PD7 register's PD7_3 bit=0 |
| | $\overline{RTS}$ output | UiC0 register's CRD bit=0<br>UiC0 register's CRS bit=1 |
| | I/O port | UiC0 register's CRD bit=1 |

**Table 10.4 P64 Pin Functions**

| Pin function | Bit set value | | | | | |
|---|---|---|---|---|---|---|
| | U1C0 register | | UCON register | | | PD6 register |
| | CRD | CRS | RCSP | CLKMD1 | CLKMD0 | PD6_4 |
| P64 | 1 | — | 0 | 0 | — | Input: 0, Output: 1 |
| $\overline{CTS}$1 | 0 | 0 | 0 | 0 | — | 0 |
| $\overline{RTS}$1 | 0 | 1 | 0 | 0 | — | — |
| $\overline{CTS}$0(Note1) | 0 | 0 | 1 | 0 | — | 0 |
| CLKS1 | — | — | — | 1(Note 2) | 1 | — |

Note 1: In addition to this, set the U0C0 register's CRD bit to "0" ($\overline{CTS}$0/$\overline{RTS}$0 enabled) and the U0 C0 register's CRS bit to "1" ($\overline{RTS}$0 selected).
Note 2: When the CLKMD1 bit = 1 and the CLKMD0 bit = 0, the following logic levels are output:
- High if the U1C0 register's CLKPOL bit = 0
- Low if the U1C0 register's CLKPOL bit = 1

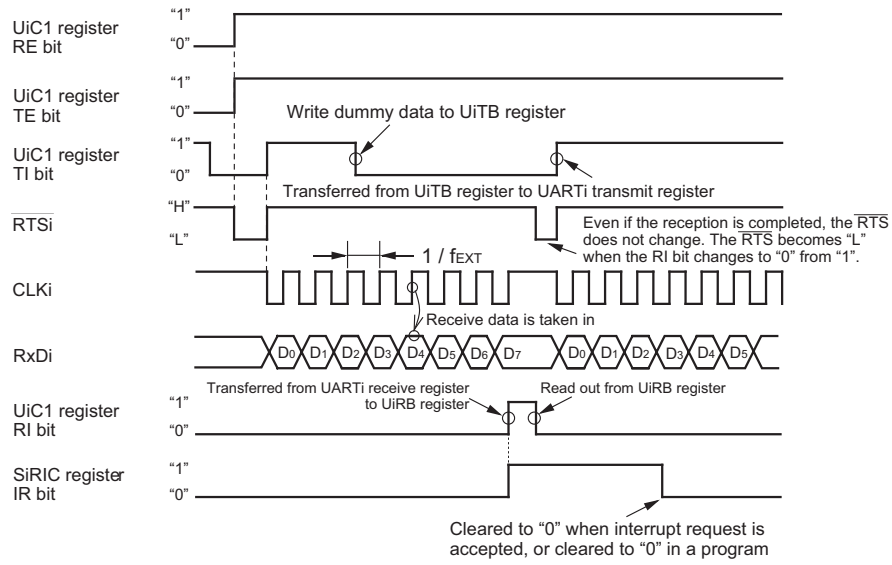
(1) Example of transmit timing (when internal clock is selected)




Tc = TCLK = 2(n + 1) / fj
fj: frequency of UiBRG count source (f1SIO, f2SIO, f8SIO, f32SIO)
n: value set to UiBRG register
i: 0 to 2

The above timing diagram applies to the case where the register bits are set as follows:
- UiMR register CKDIR bit = 0 (internal clock)
- UiC0 register CRD bit = 0 (CTS/RTS enabled), CRS bit = 0 (CTS selected)
- UiC0 register CKPOL bit = 0 (transmit data output at the falling edge and receive data taken in at the rising edge of the transfer clock)
- UiRS bit = 0 (an interrupt request occurs when the transmit buffer becomes empty): U0IRS bit is the UCON register bit 0, U1IRS bit is the UCON register bit 1, and U2IRS bit is the U2C1 register bit 4

(2) Example of receive timing (when external clock is selected)




The above timing diagram applies to the case where the register bits are set as follows:
- UiMR register CKDIR bit = 1 (external clock)
- UiC0 register CRD bit = 0 (CTS/RTS enabled), CRS bit = 1 (RTS selected)
- UiC0 register CKPOL bit = 0 (transmit data output at the falling edge and receive data taken in at the rising edge of the transfer clock)

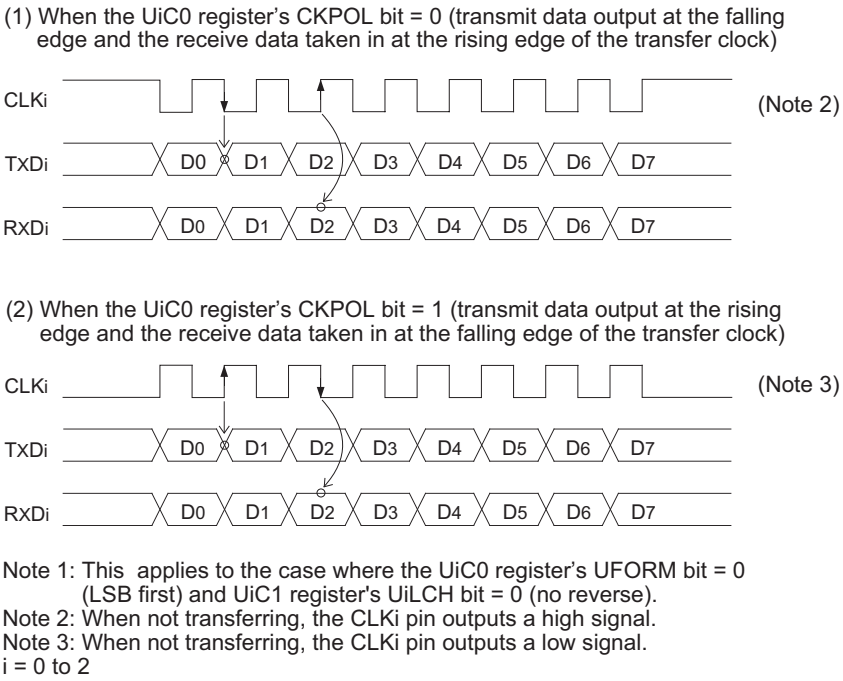
Make sure the following conditions are met when input to the CLKi pin before receiving data is high:
- UiC1 register TE bit = 1 (transmit enabled)
- UiC1 register RE bit = 1 (Receive enabled)
- Write dummy data to the UiTB register

fEXT: frequency of external clock

**Figure 10.9 Transmit and Receive Operation**

### 10.2.1 CLK Polarity Select Function

Use the UiC0 register (i = 0 to 2)'s CKPOL bit to select the transfer clock polarity. Figure 10.10 shows the polarity of the transfer clock.

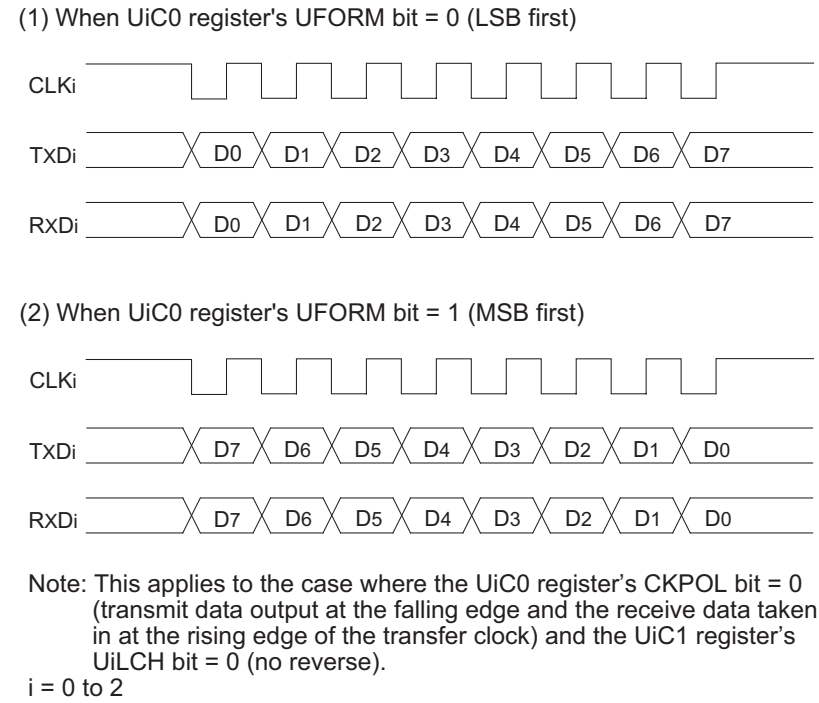



Note 1: This applies to the case where the UiC0 register's UFORM bit = 0 (LSB first) and UiC1 register's UiLCH bit = 0 (no reverse).
Note 2: When not transferring, the CLKi pin outputs a high signal.
Note 3: When not transferring, the CLKi pin outputs a low signal.
i = 0 to 2

**Figure 10.10 Transfer Clock Polarity**

### 10.2.2 LSB First/MSB First Select Function

Use the UiC0 register (i = 0 to 2)'s UFORM bit to select the transfer format. Figure 10.11 shows the transfer format.




Note: This applies to the case where the UiC0 register's CKPOL bit = 0 (transmit data output at the falling edge and the receive data taken in at the rising edge of the transfer clock) and the UiC1 register's UiLCH bit = 0 (no reverse).
i = 0 to 2

**Figure 10.11 Transfer Format**

### 10.2.3 Continuous Receive Mode

When the UiRRM bit (i = 0 to 2) = 1 (continuous receive mode), the UiC1 register's TI bit is set to "0" (data present in the UiTB register) by reading the UiRB register. In this case, i.e., UiRRM bit = 1, do not write dummy data to the UiTB register in a program. The U0RRM and U1RRM bits are the UCON register bit 2 and bit 3, respectively, and the U2RRM bit is the U2C1 register bit 5.

### 10.2.4 Serial Data Logic Switching Function

When the UiC1 register (i = 0 to 2)'s UiLCH bit = 1 (reverse), the data written to the UiTB register has its logic reversed before being transmitted. Similarly, the received data has its logic reversed when read from the UiRB register. Figure 10.12 shows serial data logic.

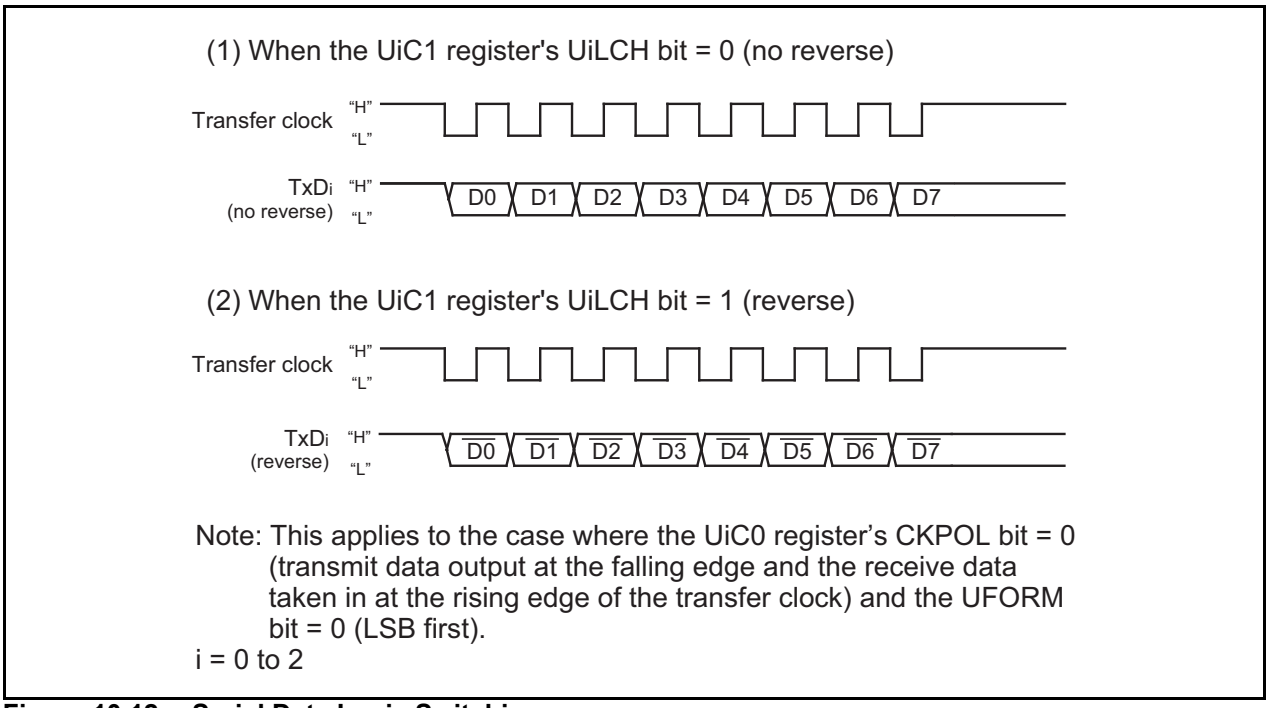



**Figure 10.12 Serial Data Logic Switching**

### 10.2.5 Transfer Clock Output From Multiple Pins (UART1)

Use the UCON register's CLKMD1 to CLKMD0 bits to select one of the two transfer clock output pins. (See Figure 10.13.) This function can be used when the selected transfer clock for UART1 is an internal clock.

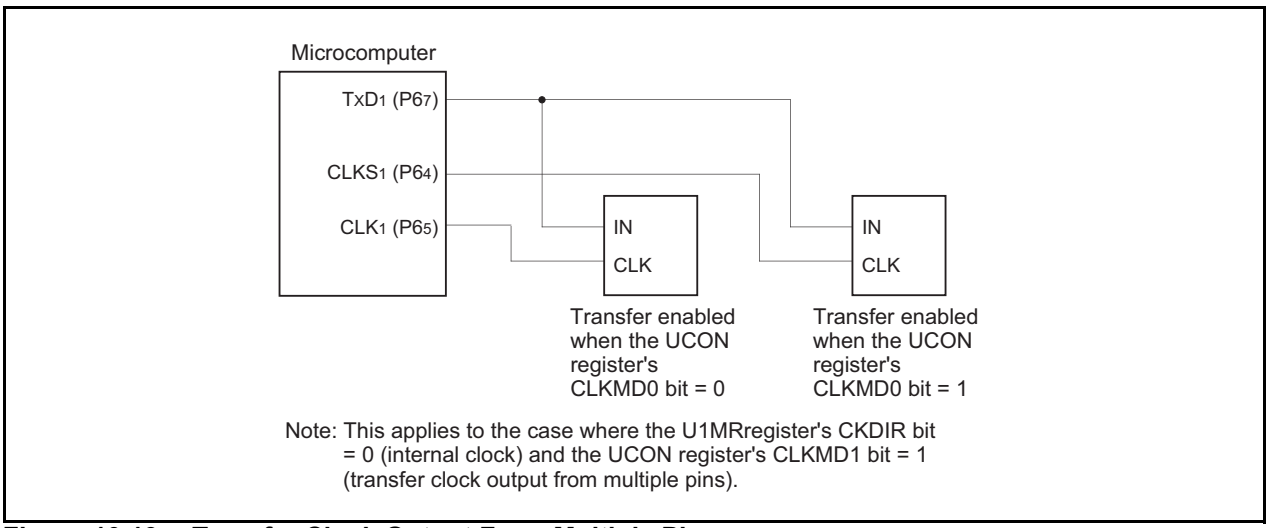



**Figure 10.13 Transfer Clock Output From Multiple Pins**

### 10.2.6 $\overline{\text{CTS}}/\overline{\text{RTS}}$ Separate Function (UART0)

This function separates $\overline{\text{CTS}}_0/\overline{\text{RTS}}_0$, outputs $\overline{\text{RTS}}_0$ from the P60 pin, and accepts as input the $\overline{\text{CTS}}_0$ from the P64 pin. To use this function, set the register bits as shown below.

- U0C0 register's CRD bit = 0 (enables UART0 $\overline{\text{CTS}}/\overline{\text{RTS}}$)
- U0C0 register's CRS bit = 1 (outputs UART0 $\overline{\text{RTS}}$)
- U1C0 register's CRD bit = 0 (enables UART1 $\overline{\text{CTS}}/\overline{\text{RTS}}$)
- U1C0 register's CRS bit = 0 (inputs UART1 $\overline{\text{CTS}}$)
- UCON register's RCSP bit = 1 (inputs $\overline{\text{CTS}}_0$ from the P64 pin)
- UCON register's CLKMD1 bit = 0 ($\text{CLKS}_1$ not used)

Note that when using the $\overline{\text{CTS}}/\overline{\text{RTS}}$ separate function, UART1 $\overline{\text{CTS}}/\overline{\text{RTS}}$ function cannot be used.

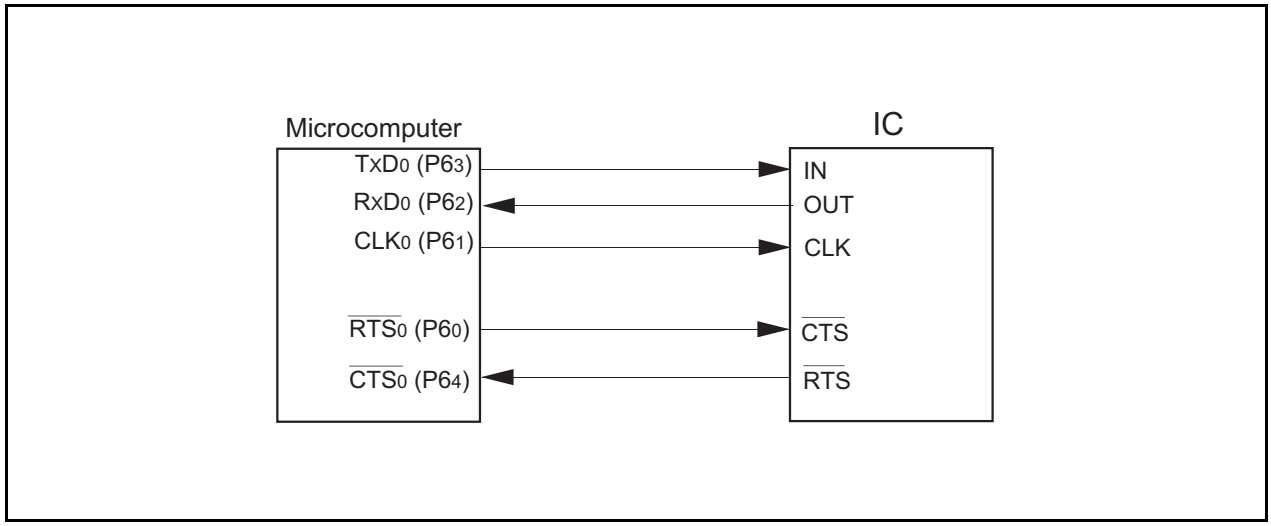



**Figure 10.14 $\overline{\text{CTS}}/\overline{\text{RTS}}$ Separat Function**

## 10.3 Clock Asynchronous Serial I/O (UART) Mode

The UART mode allows transmitting and receiving data after setting the desired transfer rate and transfer data format. Tables 10.5 lists the specifications of the UART mode.

**Table 10.5 UART Mode Specifications**

| Item | Specification |
|---|---|
| Transfer data format | • Character bit (transfer data): Selectable from 7, 8 or 9 bits<br>• Start bit: 1 bit<br>• Parity bit: Selectable from odd, even, or none<br>• Stop bit: Selectable from 1 or 2 bits |
| Transfer clock | • UiMR(i=0 to 2) register's CKDIR bit = 0 (internal clock) : fj/ 16(n+1)<br>fj = $f_{1SIO}$, $f_{2SIO}$, $f_{8SIO}$, $f_{32SIO}$. n: Setting value of UiBRG register $00_{16}$ to $FF_{16}$<br>• CKDIR bit = "1" (external clock) : $f_{EXT}$/16(n+1)<br>$f_{EXT}$: Input from CLKi pin. n :Setting value of UiBRG register $00_{16}$ to $FF_{16}$ |
| Transmission, reception control | • Selectable from $\overline{CTS}$ function, $\overline{RTS}$ function or $\overline{CTS}/\overline{RTS}$ function disable |
| Transmission start condition | • Before transmission can start, the following requirements must be met<br>– The TE bit of UiC1 register= 1 (transmission enabled)<br>– The TI bit of UiC1 register = 0 (data present in UiTB register)<br>– If $\overline{CTS}$ function is selected, input on the $\overline{CTS}$i pin = "L" |
| Reception start condition | • Before reception can start, the following requirements must be met<br>– The RE bit of UiC1 register= 1 (reception enabled)<br>– Start bit detection |
| Interrupt request generation timing | • For transmission, one of the following conditions can be selected<br>– The UiIRS bit (Note 2) = 0 (transmit buffer empty): when transferring data from the UiTB register to the UARTi transmit register (at start of transmission)<br>– The UiIRS bit =1 (transfer completed): when the serial I/O finished sending data from the UARTi transmit register<br>• For reception<br>When transferring data from the UARTi receive register to the UiRB register (at completion of reception) |
| Error detection | • Overrun error (Note 1)<br>This error occurs if the serial I/O started receiving the next data before reading the UiRB register and received the bit one before the last stop bit of the next data<br>• Framing error<br>This error occurs when the number of stop bits set is not detected<br>• Parity error<br>This error occurs when if parity is enabled, the number of 1's in parity and character bits does not match the number of 1's set<br>• Error sum flag<br>This flag is set (= 1) when any of the overrun, framing, and parity errors is encountered |
| Select function | • LSB first, MSB first selection<br>Whether to start sending/receiving data beginning with bit 0 or beginning with bit 7 can be selected<br>• Serial data logic switch<br>This function reverses the logic of the transmit/receive data. The start and stop bits are not reversed.<br>• TxD, RxD I/O polarity switch<br>This function reverses the polarities of hte TxD pin output and RxD pin input. The logic levels of all I/O data is reversed.<br>• Separate $\overline{CTS}/\overline{RTS}$ pins (UART0)<br>$\overline{CTS_0}$ and $\overline{RTS_0}$ are input/output from separate pins |

Note 1: If an overrun error occurs, the value of UiRB register will be indeterminate. The IR bit of SiRIC register does not change.
Note 2: The U0IRS and U1IRS bits respectively are the UCON register bits 0 and 1; the U2IRS bit is the U2C1 register bit 4.

**Table 10.6 Registers to Be Used and Settings in UART Mode**

| Register | Bit | Function |
|---|---|---|
| UiTB | 0 to 8 | Set transmission data (Note 1) |
| UiRB | 0 to 8 | Reception data can be read (Note 1) |
| | OER,FER,PER,SUM | Error flag |
| UiBRG | 0 to 7 | Set a transfer rate |
| UiMR | SMD2 to SMD0 | Set these bits to '100$_2$' when transfer data is 7 bits long<br>Set these bits to '101$_2$' when transfer data is 8 bits long<br>Set these bits to '110$_2$' when transfer data is 9 bits long |
| | CKDIR | Select the internal clock or external clock |
| | STPS | Select the stop bit |
| | PRY, PRYE | Select whether parity is included and whether odd or even |
| | IOPOL | Select the TxD/RxD input/output polarity |
| UiC0 | CLK0, CLK1 | Select the count source for the UiBRG register |
| | CRS | Select $\overline{\text{CTS}}$ or $\overline{\text{RTS}}$ to use |
| | TXEPT | Transmit register empty flag |
| | CRD | Enable or disable the $\overline{\text{CTS}}$ or $\overline{\text{RTS}}$ function |
| | NCH | Select TxDi pin output mode (Note 3) |
| | CKPOL | Set to "0" |
| | UFORM | LSB first or MSB first can be selected when transfer data is 8 bits long. Set this bit to "0" when transfer data is 7 or 9 bits long. |
| UiC1 | TE | Set this bit to "1" to enable transmission |
| | TI | Transmit buffer empty flag |
| | RE | Set this bit to "1" to enable reception |
| | RI | Reception complete flag |
| | U2IRS (Note 2) | Select the source of UART2 transmit interrupt |
| | U2RRM (Note 2) | Set to "0" |
| | UiLCH | Set this bit to "1" to use inverted data logic |
| | UiERE | Set to "0" |
| UiSMR | 0 to 7 | Set to "0" |
| UiSMR2 | 0 to 7 | Set to "0" |
| UiSMR3 | 0 to 7 | Set to "0" |
| UiSMR4 | 0 to 7 | Set to "0" |
| UCON | U0IRS, U1IRS | Select the source of UART0/UART1 transmit interrupt |
| | U0RRM, U1RRM | Set to "0" |
| | CLKMD0 | Invalid because CLKMD1 = 0 |
| | CLKMD1 | Set to "0" |
| | RCSP | Set this bit to "1" to accept as input the UART0 $\overline{\text{CTS}_0}$ signal from the P6$_4$ pin |
| | 7 | Set to "0" |

Note 1: The bits used for transmit/receive data are as follows: Bit 0 to bit 6 when transfer data is 7 bits long; bit 0 to bit 7 when transfer data is 8 bits long; bit 0 to bit 8 when transfer data is 9 bits long.

Note 2: Set the U0C1 and U1C1 registers bit 4 to bit 5 to "0". The U0IRS, U1IRS, U0RRM and U1RRM bits are included in the UCON register.

Note 3: TxD2 pin is N channel open-drain output. Set the U2C0 register's NCH bit to "0".

i=0 to 2

Table 10.7 lists the functions of the input/output pins during UART mode. Table 10.8 lists the P64 pin functions during UART mode. Note that for a period from when the UARTi operation mode is selected to when transfer starts, the TxDi pin outputs an "H". (If the N-channel open-drain output is selected, this pin is in a high-impedance state.)

**Table 10.7 I/O Pin Functions**

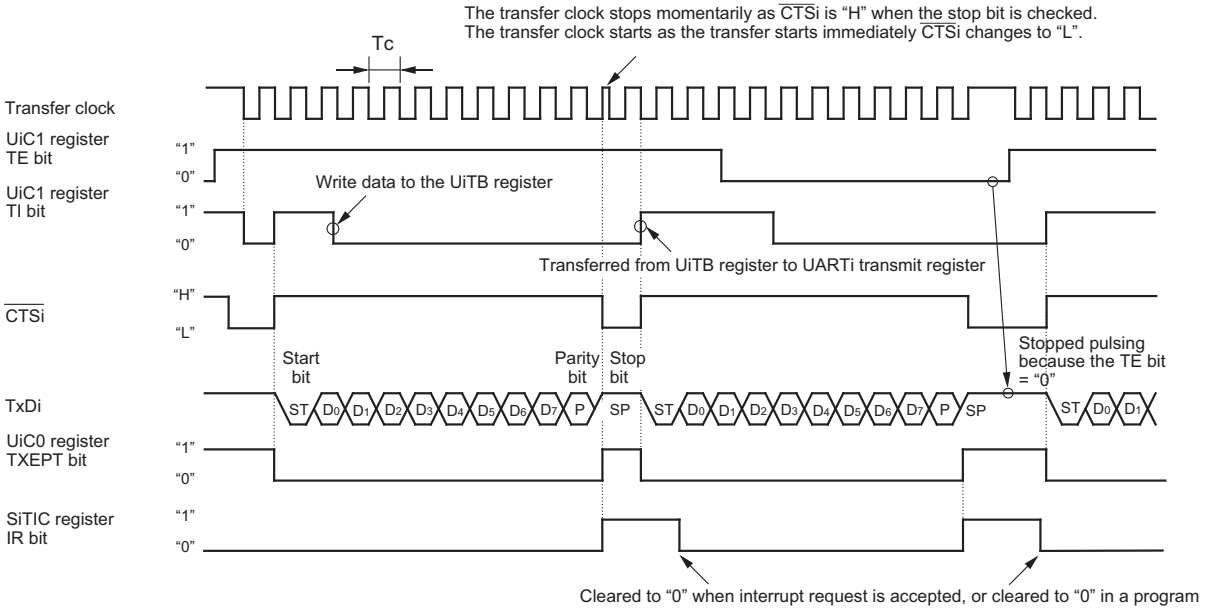
| Pin name | Function | Method of selection |
|---|---|---|
| TxDi (i = 0 to 2) (P63, P67, P70) | Serial data output | (Outputs dummy data when performing reception only) |
| RxDi (P62, P66, P71) | Serial data input | PD6 register's PD6_2 bit=0, PD6_6 bit=0, PD7 register's PD7_1 bit=0<br>(Can be used as an input port when performing transmission only) |
| CLKi (P61, P65, P72) | Input/output port | UiMR register's CKDIR bit=0 |
| | Transfer clock input | UiMR register's CKDIR bit=1<br>PD6 register's PD6_1 bit=0, PD6_5 bit=0, PD7 register's PD7_2 bit=0 |
| $\overline{\text{CTS}}$i/$\overline{\text{RTS}}$i (P60, P64, P73) | $\overline{\text{CTS}}$ input | UiC0 register's CRD bit=0<br>UiC0 register's CRS bit=0<br>PD6 register's PD6_0 bit=0, PD6_4 bit=0, PD7 register's PD7_3 bit=0 |
| | $\overline{\text{RTS}}$ output | UiC0 register's CRD bit=0<br>UiC0 register's CRS bit=1 |
| | Input/output port | UiC0 register's CRD bit=1 |

**Table 10.8 P64 Pin Functions**

| Pin function | Bit set value | | | | |
|---|---|---|---|---|---|
| | U1C0 register | | UCON register | | PD6 register |
| | CRD | CRS | RCSP | CLKMD1 | PD6_4 |
| P64 | 1 | — | 0 | 0 | Input: 0, Output: 1 |
| $\overline{\text{CTS}}$1 | 0 | 0 | 0 | 0 | 0 |
| $\overline{\text{RTS}}$1 | 0 | 1 | 0 | 0 | — |
| $\overline{\text{CTS}}$0 (Note) | 0 | 0 | 1 | 0 | 0 |

Note: In addition to this, set the U0C0 register's CRD bit to "0" ($\overline{\text{CTS}}$0/$\overline{\text{RTS}}$0 enabled) and the U0C0 register's CRS bit to "1" ($\overline{\text{RTS}}$0 selected).

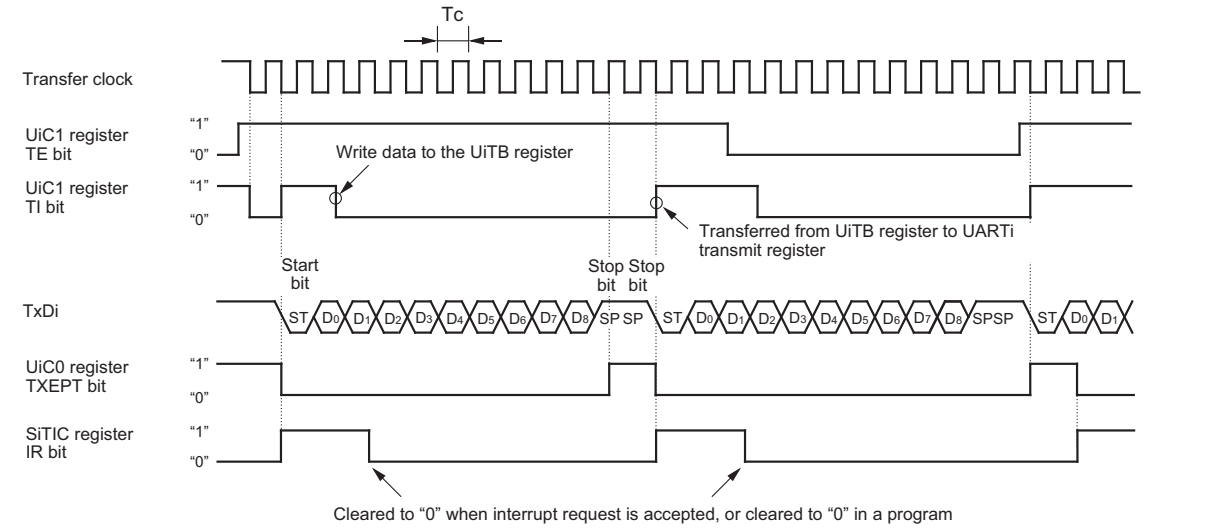
(1) Example of transmit timing when transfer data is 8 bits long (parity enabled, one stop bit)




The above timing diagram applies to the case where the register bits are set as follows:
- UiMR register PRYE bit = 1 (parity enabled)
- UiMR register STPS bit = 0 (1 stop bit)
- UiC0 register CRD bit = 0 (CTS/RTS enabled), CRS bit = 0 (CTS selected)
- UiRS bit = 1 (an interrupt request occurs when transmit completed):
  U0IRS bit is the UCON register bit 0, U1IRS bit is the UCON register bit 1, and U2IRS bit is the U2C1 register bit 4

Tc = 16 (n + 1) / fj or 16 (n + 1) / fEXT
fj : frequency of UiBRG count source (f1SIO, f2SIO, f8SIO, f32SIO)
fEXT : frequency of UiBRG count source (external clock)
n : value set to UiBRG
i: 0 to 2

(2) Example of transmit timing when transfer data is 9 bits long (parity disabled, two stop bits)




The above timing diagram applies to the case where the register bits are set as follows:
- UiMR register PRYE bit = 0 (parity disabled)
- UiMR register STPS bit = 1 (2 stop bits)
- UiC0 register CRD bit = 1 (CTS/RTS disabled)
- UiRS bit = 0 (an interrupt request occurs when transmit buffer becomes empty):
  U0IRS bit is the UCON register bit 0, U1IRS bit is the UCON register bit 1, and U2IRS bit is the U2C1 register bit 4

Tc = 16 (n + 1) / fj or 16 (n + 1) / fEXT
fj : frequency of UiBRG count source (f1SIO, f2SIO, f8SIO, f32SIO)
fEXT : frequency of UiBRG count source (external clock)
n : value set to UiBRG
i: 0 to 2

**Figure 10.15 Transmit Operation**

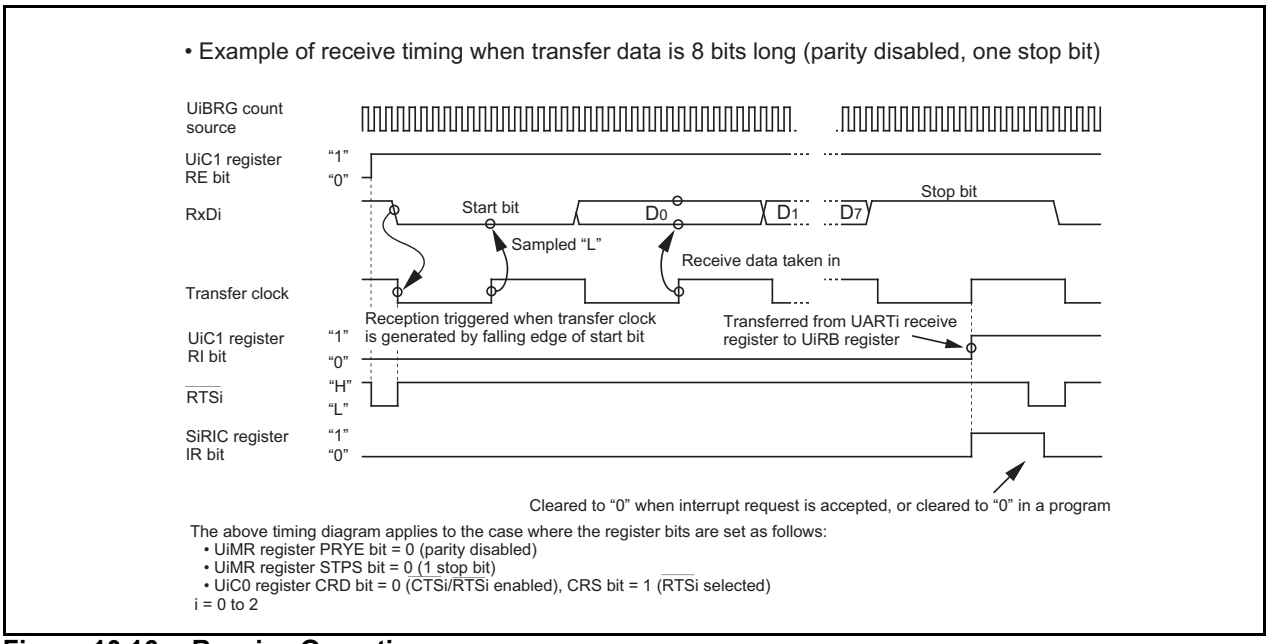
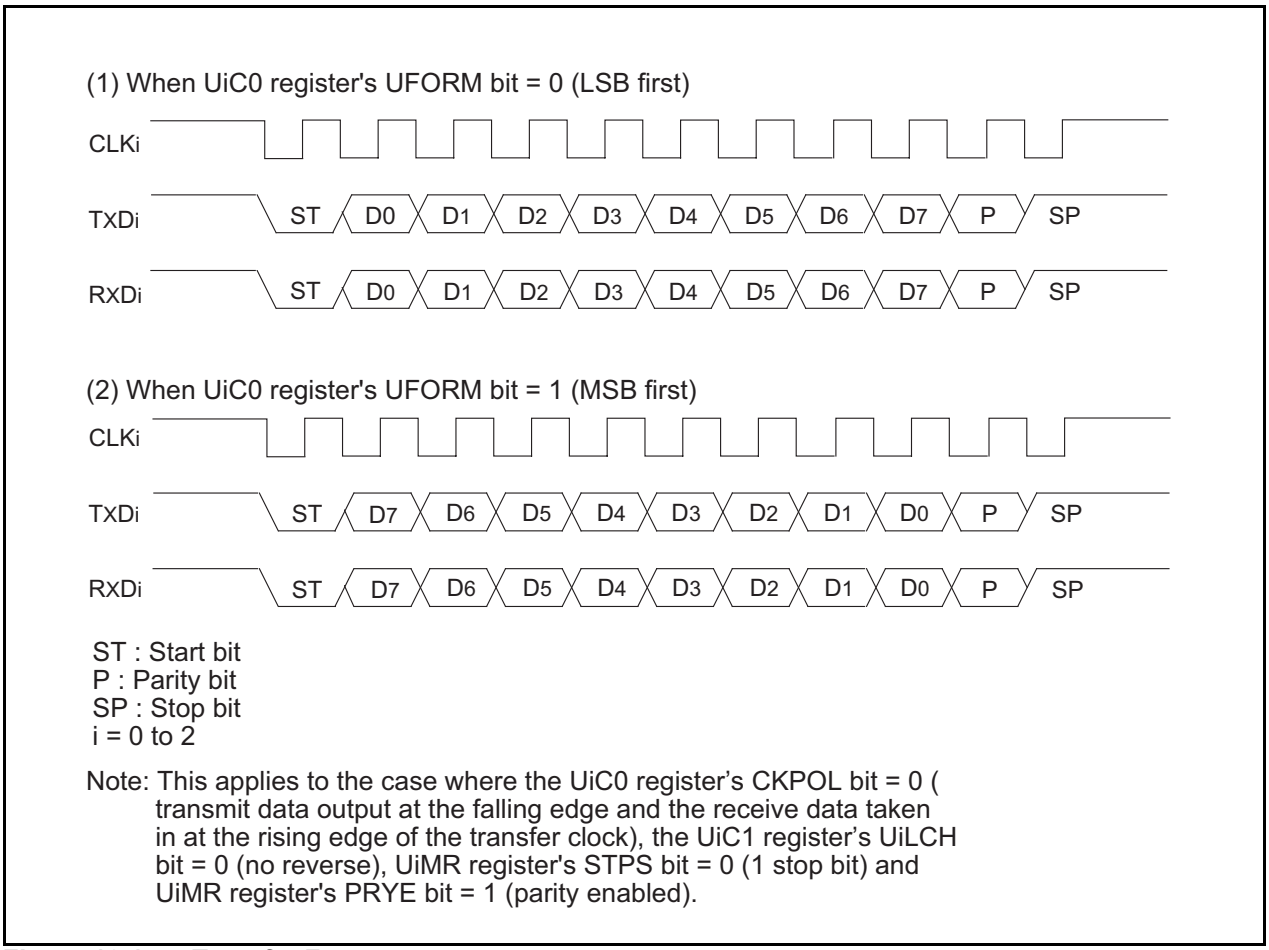



**Figure 10.16 Receive Operation**

### 10.3.1 LSB First/MSB First Select Function

As shown in Figure 10.17, use the UiC0 register's UFORM bit to select the transfer format. This function is valid when transfer data is 8 bits long.

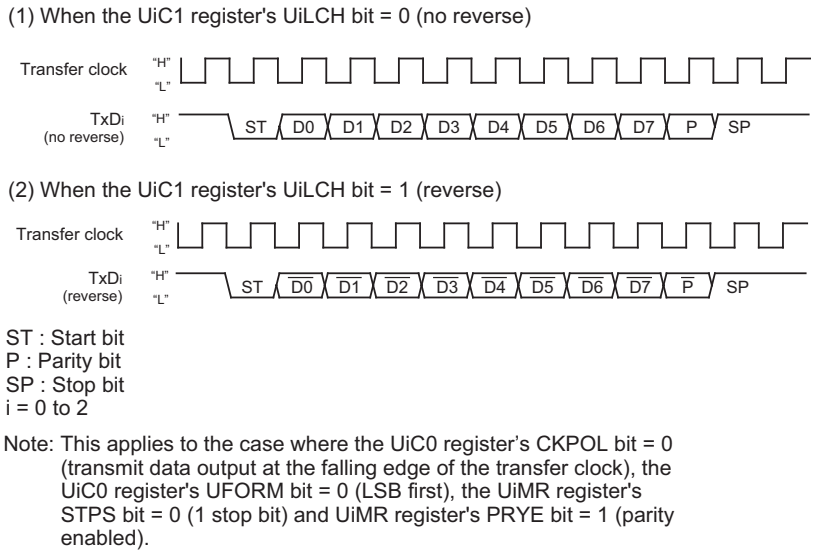



Note: This applies to the case where the UiC0 register's CKPOL bit = 0 ( transmit data output at the falling edge and the receive data taken in at the rising edge of the transfer clock), the UiC1 register's UiLCH bit = 0 (no reverse), UiMR register's STPS bit = 0 (1 stop bit) and UiMR register's PRYE bit = 1 (parity enabled).

**Figure 10.17 Transfer Format**

### 10.3.2 Serial Data Logic Switching Function

The data written to the UiTB register has its logic reversed before being transmitted. Similarly, the received data has its logic reversed when read from the UiRB register. Figure 10.18 shows serial data logic.



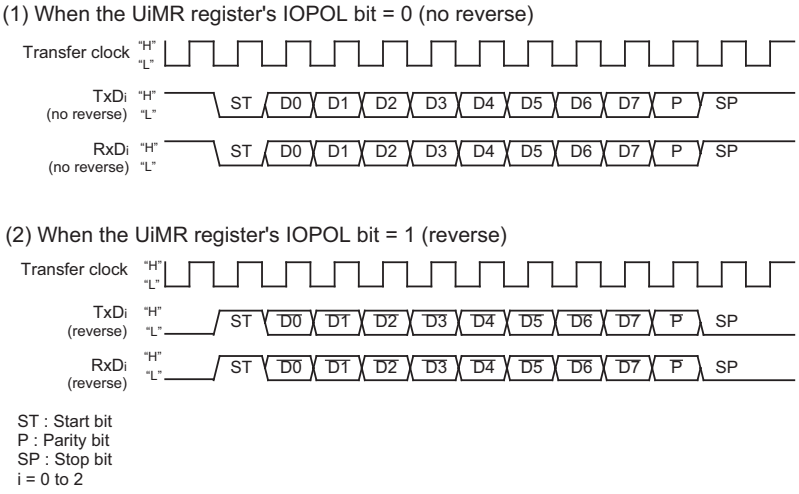

ST : Start bit
P : Parity bit
SP : Stop bit
i = 0 to 2

Note: This applies to the case where the UiC0 register's CKPOL bit = 0 (transmit data output at the falling edge of the transfer clock), the UiC0 register's UFORM bit = 0 (LSB first), the UiMR register's STPS bit = 0 (1 stop bit) and UiMR register's PRYE bit = 1 (parity enabled).

**Figure 10.18 Serial Data Logic Switching**

### 10.3.3 TxD and RxD I/O Polarity Inverse Function

This function inverses the polarities of the TXDi pin output and RXDi pin input. The logic levels of all input/output data (including the start, stop and parity bits) are inversed. Figure 10.19 shows the TXD pin output and RXD pin input polarity inverse.




ST : Start bit
P : Parity bit
SP : Stop bit
i = 0 to 2

Note: This applies to the case where the UiC0 register's UFORM bit = 0 (LSB first), the UiMR register's STPS bit = 0 (1 stop bit) and the UiMR register's PRYE bit = 1 (parity enabled).

**Figure 10.19 TxD and RxD I/O Polarity Inverse**

### 10.3.4 $\overline{\text{CTS}}/\overline{\text{RTS}}$ Separate Function (UART0)

This function separates $\overline{\text{CTS}}_0/\overline{\text{RTS}}_0$, outputs $\overline{\text{RTS}}_0$ from the $P6_0$ pin, and accepts as input the $\overline{\text{CTS}}_0$ from the $P6_4$ pin. To use this function, set the register bits as shown below.

- U0C0 register's CRD bit = 0 (enables UART0 $\overline{\text{CTS}}/\overline{\text{RTS}}$)
- U0C0 register's CRS bit = 1 (outputs UART0 $\overline{\text{RTS}}$)
- U1C0 register's CRD bit = 0 (enables UART1 $\overline{\text{CTS}}/\overline{\text{RTS}}$)
- U1C0 register's CRS bit = 0 (inputs UART1 $\overline{\text{CTS}}$)
- UCON register's RCSP bit = 1 (inputs $\text{CTS}_0$ from the $P6_4$ pin)
- UCON register's CLKMD1 bit = 0 ($\text{CLKS}_1$ not used)

Note that when using the $\overline{\text{CTS}}/\overline{\text{RTS}}$ separate function, UART1 $\overline{\text{CTS}}/\overline{\text{RTS}}$ function cannot be used.

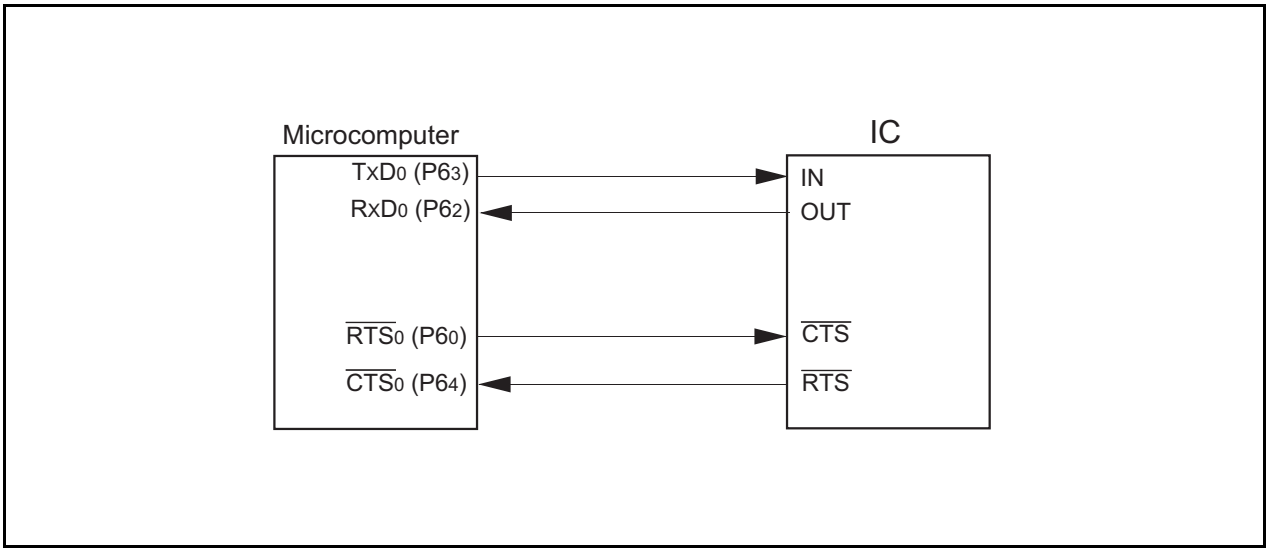



**Figure 10.20 $\overline{\text{CTS}}/\overline{\text{RTS}}$ Separate Function**

## 10.4 Special Mode 1 (I²C mode)

I²C mode is provided for use as a simplified I²C interface compatible mode. Table 10.9 lists the specifications of the I²C mode. Table 10.10 to 10.11 lists the registers used in the I²C mode and the register values set, Table 10.12 lists the I²C mode functions. Figure 10.21 shows the block diagram for I²C mode. Figure 10.22 shows SCLi timing.

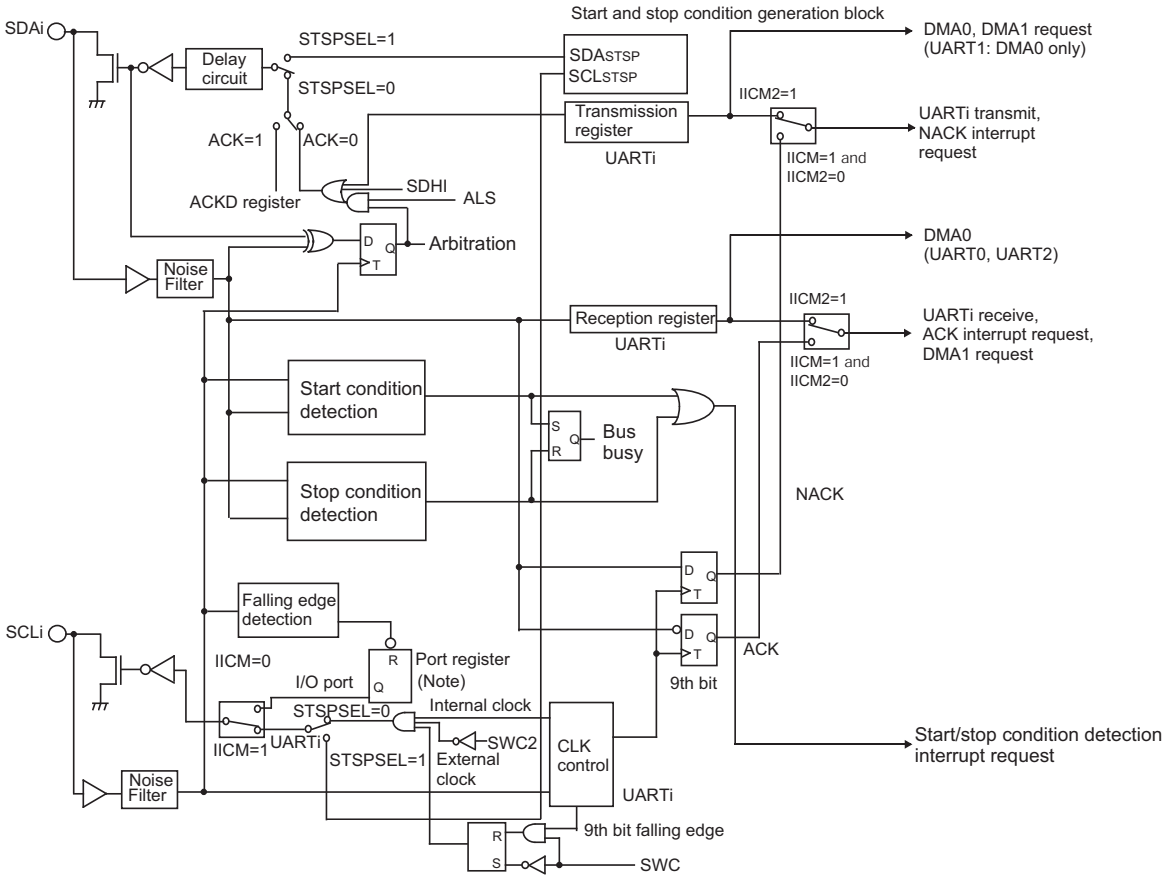
As shown in Table 10.12, the microcomputer is placed in I²C mode by setting the SMD2 to SMD0 bits to '010$_2$' and the IICM bit to "1". Because SDAi transmit output has a delay circuit attached, SDAi output does not change state until SCLi goes low and remains stably low.

**Table 10.9 I²C Mode Specifications**

| Item | Specification |
|---|---|
| Transfer data format | • Transfer data length: 8 bits |
| Transfer clock | • During master<br>UiMR(i=0 to 2) register's CKDIR bit = "0" (internal clock) : fj/ 2(n+1)<br>fj = $f_{1SIO}$, $f_{2SIO}$, $f_{8SIO}$, $f_{32SIO}$. n: Setting value of UiBRG register $00_{16}$ to $FF_{16}$<br>• During slave<br>CKDIR bit = "1" (external clock) : Input from SCLi pin |
| Transmission start condition | • Before transmission can start, the following requirements must be met (Note 1)<br>– The TE bit of UiC1 register= 1 (transmission enabled)<br>– The TI bit of UiC1 register = 0 (data present in UiTB register) |
| Reception start condition | • Before reception can start, the following requirements must be met (Note 1)<br>– The RE bit of UiC1 register= 1 (reception enabled)<br>– The TE bit of UiC1 register= 1 (transmission enabled)<br>– The TI bit of UiC1 register= 0 (data present in the UiTB register) |
| Interrupt request generation timing | When start or stop condition is detected, acknowledge undetected, and acknowledge detected |
| Error detection | • Overrun error (Note 2)<br>This error occurs if the serial I/O started receiving the next data before reading the UiRB register and received the 8th bit of the next data |
| Select function | • Arbitration lost<br>Timing at which the UiRB register's ABT bit is updated can be selected<br>• SDAi digital delay<br>No digital delay or a delay of 2 to 8 UiBRG count source clock cycles selectable<br>• Clock phase setting<br>With or without clock delay selectable |

Note 1: When an external clock is selected, the conditions must be met while the external clock is in the high state.

Note 2: If an overrun error occurs, the value of UiRB register will be indeterminate. The IR bit of SiRIC register does not change.




This diagram applies to the case where the UiMR register's SMD2 to SMD0 bits = $010_2$ and the UiSMR register's IICM bit = 1.

IICM : UiSMR register bit
IICM2, SWC, ALS, SWC2, SDHI : UiSMR2 register bit
STSPSEL, ACKD, ACKC : UiSMR4 register bit

i=0 to 2

Note: If the IICM bit = 1, the pin can be read even when the PD6_2, PD6_6 or PD7_1 bit = 1 (output mode).

**Figure 10.21 $I^2C$ Mode Block Diagram**

**Table 10.10 Registers to Be Used and Settings in I2C Mode (1) (Continued)**

| Register | Bit | Function | |
|---|---|---|---|
| | | Master | Slave |
| UiTB (Note 3) | 0 to 7 | Set transmission data | Set transmission data |
| UiRB (Note 3) | 0 to 7 | Reception data can be read | Reception data can be read |
| | 8 | ACK or NACK is set in this bit | ACK or NACK is set in this bit |
| | ABT | Arbitration lost detection flag | Invalid |
| | OER | Overrun error flag | Overrun error flag |
| UiBRG | 0 to 7 | Set a transfer rate | Invalid |
| UiMR (Note 3) | SMD2 to SMD0 | Set to '010$_2$' | Set to '010$_2$' |
| | CKDIR | Set to "0" | Set to "1" |
| | IOPOL | Set to "0" | Set to "0" |
| UiC0 | CLK1, CLK0 | Select the count source for the UiBRG register | Invalid |
| | CRS | Invalid because CRD = 1 | Invalid because CRD = 1 |
| | TXEPT | Transmit buffer empty flag | Transmit buffer empty flag |
| | CRD | Set to "1" | Set to "1" |
| | NCH | Set to "1" (Note 2) | Set to "1" (Note 2) |
| | CKPOL | Set to "0" | Set to "0" |
| | UFORM | Set to "1" | Set to "1" |
| UiC1 | TE | Set this bit to "1" to enable transmission | Set this bit to "1" to enable transmission |
| | TI | Transmit buffer empty flag | Transmit buffer empty flag |
| | RE | Set this bit to "1" to enable reception | Set this bit to "1" to enable reception |
| | RI | Reception complete flag | Reception complete flag |
| | U2IRS (Note 1) | Invalid | Invalid |
| | U2RRM (Note 1), UiLCH, UiERE | Set to "0" | Set to "0" |
| UiSMR | IICM | Set to "1" | Set to "1" |
| | ABC | Select the timing at which arbitration-lost is detected | Invalid |
| | BBS | Bus busy flag | Bus busy flag |
| | 3 to 7 | Set to "0" | Set to "0" |
| UiSMR2 | IICM2 | Refer to Table 11.12 | Refer to Table 11.12 |
| | CSC | Set this bit to "1" to enable clock synchronization | Set to "0" |
| | SWC | Set this bit to "1" to have SCLi output fixed to "L" at the falling edge of the 9th bit of clock | Set this bit to "1" to have SCLi output fixed to "L" at the falling edge of the 9th bit of clock |
| | ALS | Set this bit to "1" to have SDAi output stopped when arbitration-lost is detected | Set to "0" |
| | STAC | Set to "0" | Set this bit to "1" to initialize UARTi at start condition detection |
| | SWC2 | Set this bit to "1" to have SCLi output forcibly pulled low | Set this bit to "1" to have SCLi output forcibly pulled low |
| | SDHI | Set this bit to "1" to disable SDAi output | Set this bit to "1" to disable SDAi output |
| | 7 | Set to "0" | Set to "0" |
| UiSMR3 | 0, 2, 4 and NODC | Set to "0" | Set to "0" |
| | CKPH | Refer to Table 11.12 | Refer to Table 11.12 |
| | DL2 to DL0 | Set the amount of SDAi digital delay | Set the amount of SDAi digital delay |

i=0 to 2

Notes:
1. Set the U0C1 and U1C1 register bit 4 and bit 5 to "0". The U0IRS, U1IRS, U0RRM and U1RRM bits are in the UCON register.
2. TxD2 pin is N channel open-drain output. Set the NCH bit in the U2C0 register to "0".
3. Not all register bits are described above. Set those bits to "0" when writing to the registers in I$^2$C mode.

**Table 10.11 Registers to Be Used and Settings in I2C Mode (2) (Continued)**

| Register | Bit | Function | |
|---|---|---|---|
| | | Master | Slave |
| UiSMR4 | STAREQ | Set this bit to “1” to generate start condition | Set to “0” |
| | RSTAREQ | Set this bit to “1” to generate restart condition | Set to “0” |
| | STPREQ | Set this bit to “1” to generate stop condition | Set to “0” |
| | STSPSEL | Set this bit to “1” to output each condition | Set to “0” |
| | ACKD | Select ACK or NACK | Select ACK or NACK |
| | ACKC | Set this bit to “1” to output ACK data | Set this bit to “1” to output ACK data |
| | SCLHI | Set this bit to “1” to have SCLi output stopped when stop condition is detected | Set to “0” |
| | SWC9 | Set to “0” | Set this bit to “1” to set the SCLi to “L” hold at the next falling edge of the 9th bit of clock |
| IFSR2A | IFSR26, ISFR27 | Set to “1” | Set to “1” |
| UCON | U0IRS, U1IRS | Invalid | Invalid |
| | 2 to 7 | Set to “0” | Set to “0” |

i=0 to 2

**Table 10.12 $I^2C$ Mode Functions**

| Function | Clock synchronous serial I/O mode (SMD2 to SMD0 = $001_2$, IICM = 0) | $I^2C$ mode (SMD2 to SMD0 = $010_2$, IICM = 1) | | | |
|---|---|---|---|---|---|
| | | IICM2 = 0 (NACK/ACK interrupt) | | IICM2 = 1 (UART transmit/ receive interrupt) | |
| | | CKPH = 0 (No clock delay) | CKPH = 1 (Clock delay) | CKPH = 0 (No clock delay) | CKPH = 1 (Clock delay) |
| Factor of interrupt number 6, 7 and 10 (Note 1, 5, 7) | ———— | Start condition detection or stop condition detection (Refer to "Table 10.13. STSPSEL Bit Functions") | | | |
| Factor of interrupt number 15, 17 and 19 (Note 1, 6) | UARTi transmission Transmission started or completed (selected by UiIRS) | No acknowledgment detection (NACK) Rising edge of SCLi 9th bit | | UARTi transmission Rising edge of SCLi 9th bit | UARTi transmission Falling edge of SCLi next to the 9th bit |
| Factor of interrupt number 16, 18 and 20 (Note 1, 6) | UARTi reception When 8th bit received CKPOL = 0 (rising edge) CKPOL = 1 (falling edge) | Acknowledgment detection (ACK) Rising edge of SCLi 9th bit | | UARTi reception Falling edge of SCLi 9th bit | |
| Timing for transferring data from the UART reception shift register to the UiRB register | CKPOL = 0 (rising edge) CKPOL = 1 (falling edge) | Rising edge of SCLi 9th bit | | Falling edge of SCLi 9th bit | Falling and rising edges of SCLi 9th bit |
| UARTi transmission output delay | Not delayed | Delayed | | | |
| Functions of $P6_3$, $P6_7$ and $P7_0$ pins | TxDi output | SDAi input/output | | | |
| Functions of $P6_2$, $P6_6$ and $P7_1$ pins | RxDi input | SCLi input/output | | | |
| Functions of $P6_1$, $P6_5$ and $P7_2$ pins | CLKi input or output selected | ———— (Cannot be used in $I^2C$ mode) | | | |
| Noise filter width | 15ns | 200ns | | | |
| Read RxDi and SCLi pin levels | Possible when the corresponding port direction bit = 0 | Always possible no matter how the corresponding port direction bit is set | | | |
| Initial value of TxDi and SDAi outputs | CKPOL = 0 (H) CKPOL = 1 (L) | The value set in the port register before setting $I^2C$ mode (Note 2) | | | |
| Initial and end values of SCLi | ———— | H | L | H | L |
| DMA1 factor (Refer to Fig 10.22) | UARTi reception | Acknowledgment detection (ACK) | | UARTi reception Falling edge of SCLi 9th bit | |
| Store received data | 1st to 8th bits are stored in UiRB register bit 0 to bit 7 | 1st to 8th bits are stored in UiRB register bit 7 to bit 0 | | 1st to 7th bits are stored in UiRB register bit 6 to bit 0, with 8th bit stored in UiRB register bit 8 | |
| | | | | | 1st to 8th bits are stored in UiRB register bit 7 to bit 0 (Note 3) |
| Read received data | UiRB register status is read directly as is | | | | Read UiRB register Bit 6 to bit 0 as bit 7 to bit 1, and bit 8 as bit 0 (Note 4) |

i = 0 to 2

Note 1: If the source or cause of any interrupt is changed, the IR bit in the interrupt control register for the changed interrupt may inadvertently be set to "1" (interrupt requested). (Refer to "precautions for interrupts" of the Usage Notes Reference Book.)
If one of the bits shown below is changed, the interrupt source, the interrupt timing, etc. change. Therefore, always be sure to clear the IR bit to "0" (interrupt not requested) after changing those bits.
SMD2 to SMD0 bits in the UiMR register, IICM bit in the UiSMR register, IICM2 bit in the UiSMR2 register, CKPH bit in the UiSMR3 register

Note 2: Set the initial value of SDAi output while the UiMR register's SMD2 to SMD0 bits = '$000_2$' (serial I/O disabled).

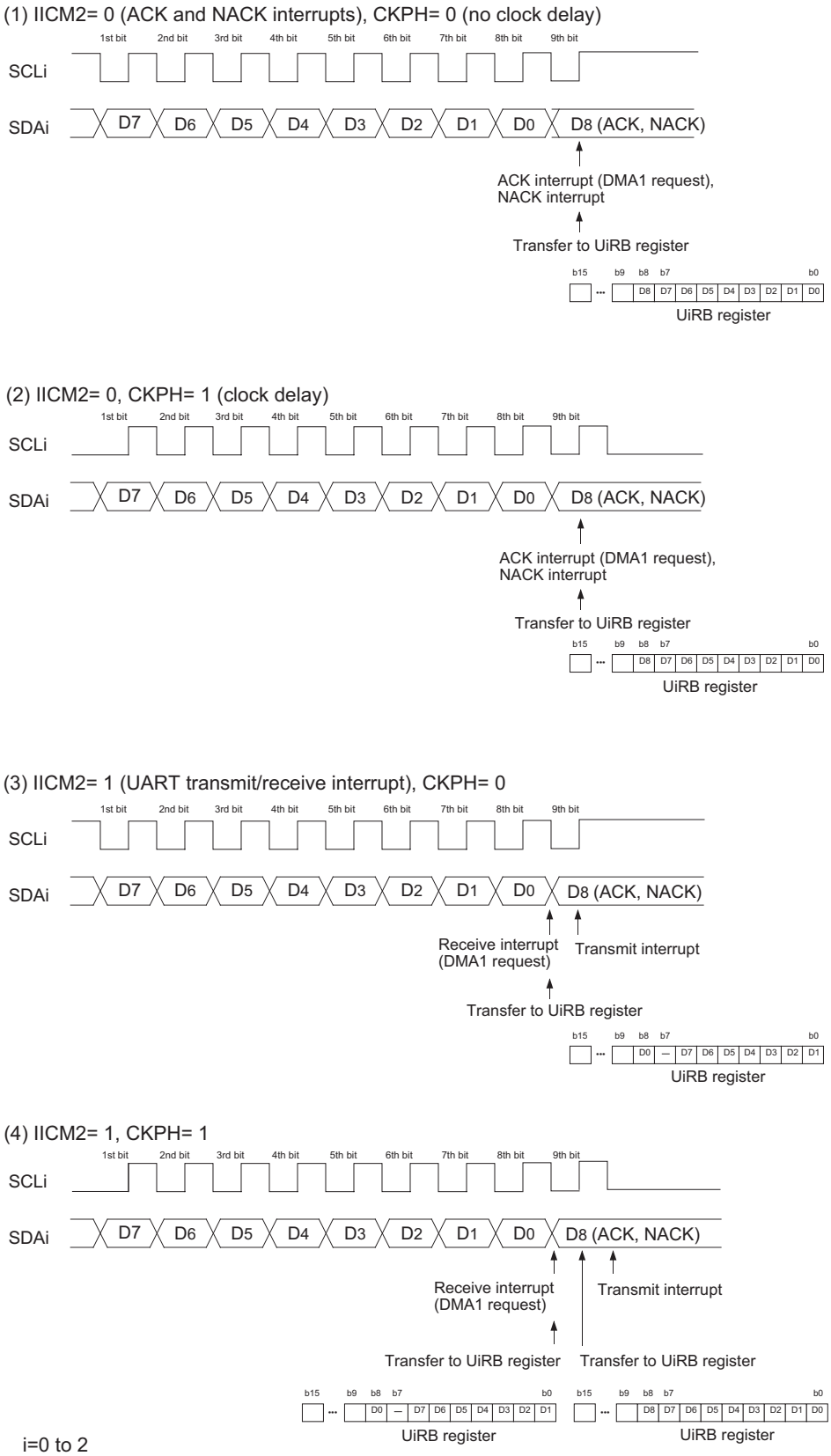
Note 3: Second data transfer to UiRB register (Rising edge of SCLi 9th bit)

Note 4: First data transfer to UiRB register (Falling edge of SCLi 9th bit)

Note 5: Refer to "Figure 10.13. STSPSEL Bit Functions".

Note 6: Refer to "Figure 10.22. Transfer to UiRB Register and Interrupt Timing" .

Note 7: When using UART0, be sure to set the IFSR26 bit in the IFSR2A register to "1" (cause of interrupt: UART0 bus collision).
When using UART1, be sure to set the IFSR27 bit in the IFSR2A register to "1" (cause of interrupt: UART1 bus collision).




i=0 to 2

This diagram applies to the case where the following condition is met.
• UiMR register CKDIR bit = 0 (Slave selected)

**Figure 10.22 Transfer to UiRB Register and Interrupt Timing**

### 10.4.1 Detection of Start and Stop Condtion

Whether a start or a stop condition has been detected is determined.
A start condition-detected interrupt request is generated when the SDAi pin changes state from high to low while the SCLi pin is in the high state. A stop condition-detected interrupt request is generated when the SDAi pin changes state from low to high while the SCLi pin is in the high state.
Because the start and stop condition-detected interrupts share the interrupt control register and vector, check the UiSMR register's BBS bit to determine which interrupt source is requesting the interrupt.

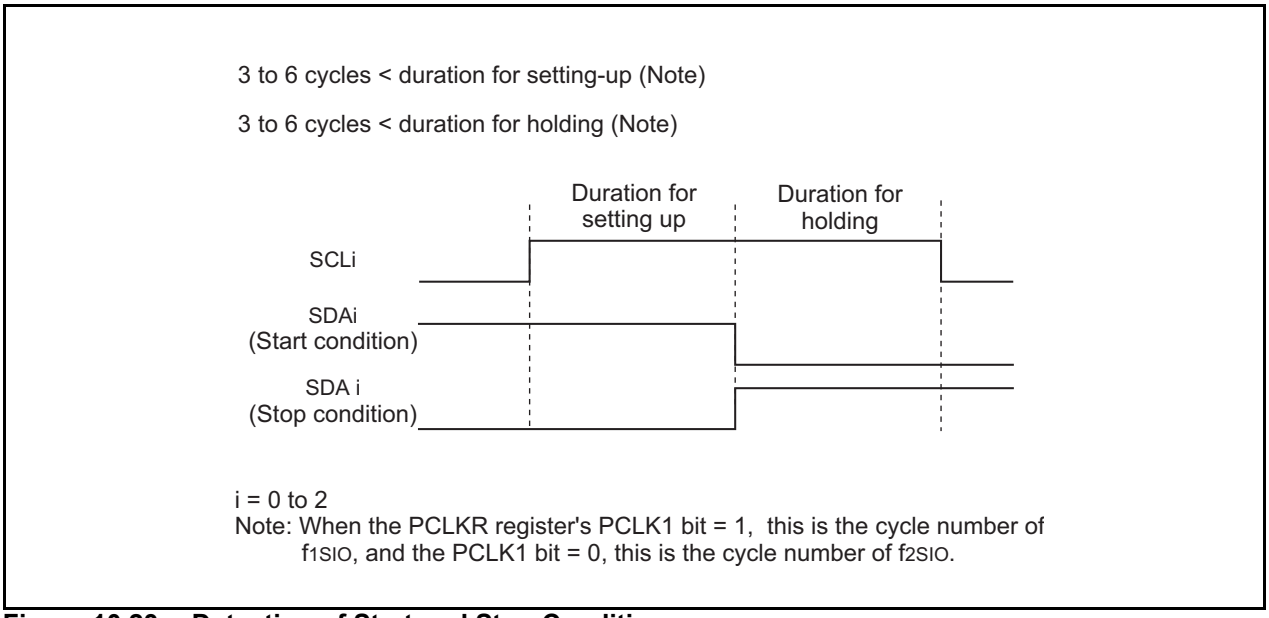



**Figure 10.23 Detection of Start and Stop Condition**

### 10.4.2 Output of Start and Stop Condition

A start condition is generated by setting the UiSMR4 register (i = 0 to 2)'s STAREQ bit to "1" (start).
A restart condition is generated by setting the UiSMR4 register's RSTAREQ bit to "1" (start).
A stop condition is generated by setting the UiSMR4 register's STPREQ bit to "1" (start).
The output procedure is described below.
Set the STAREQ bit, RSTAREQ bit or STPREQ bit to "1" (start).
Set the STSPSEL bit in the UiSMR4 register to "1" (output).
The function of the STSPSEL bit is shown in Table 10.13 and Figure 10.24.

**Table 10.13 STSPSEL Bit Functions**

| Function | STSPSEL = 0 | STSPSEL = 1 |
|---|---|---|
| Output of SCLi and SDAi pins | Output of transfer clock and data<br>Output of start/stop condition is accomplished by a program using ports (not automatically generated in hardware) | Output of a start/stop condition according to the STAREQ, RSTAREQ and STPREQ bit |
| Start/stop condition interrupt request generation timing | Start/stop condition detection | Finish generating start/stop condition |

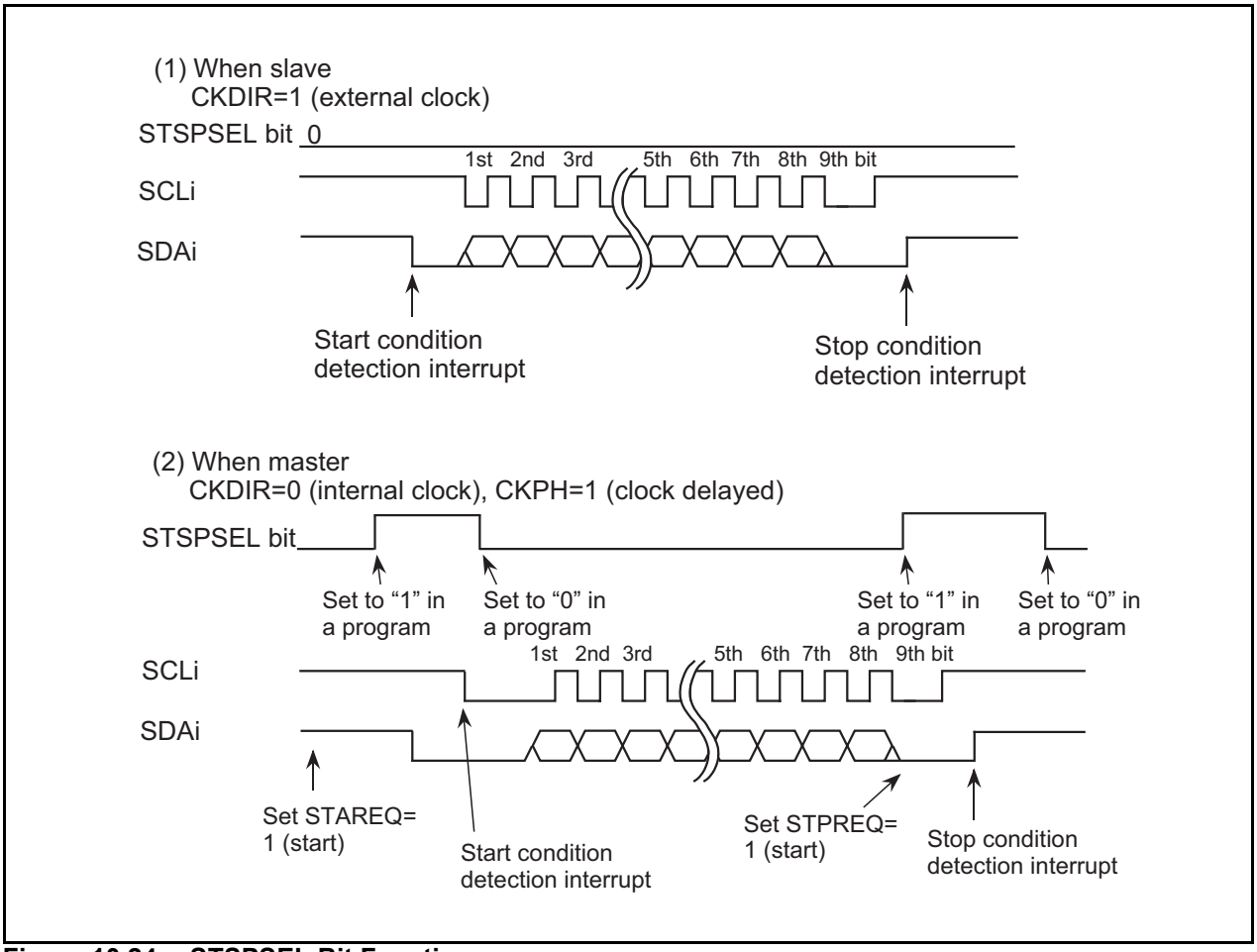



**Figure 10.24 STSPSEL Bit Functions**

## 10.4.3 Arbitration

Unmatching of the transmit data and SDAi pin input data is checked synchronously with the rising edge of SCLi. Use the UiSMR register's ABC bit to select the timing at which the UiRB register's ABT bit is updated. If the ABC bit = 0 (updated bitwise), the ABT bit is set to "1" at the same time unmatching is detected during check, and is cleared to "0" when not detected. In cases when the ABC bit is set to "1", if unmatching is detected even once during check, the ABT bit is set to "1" (unmatching detected) at the falling edge of the clock pulse of 9th bit. If the ABT bit needs to be updated bytewise, clear the ABT bit to "0" (undetected) after detecting acknowledge in the first byte, before transferring the next byte.

Setting the UiSMR2 register's ALS bit to "1" (SDA output stop enabled) causes arbitration-lost to occur, in which case the SDAi pin is placed in the high-impedance state at the same time the ABT bit is set to "1" (unmatching detected).

### 10.4.4 Transfer Clock

Data is transmitted/received using a transfer clock like the one shown in Figure 10.24.
The UiSMR2 register's CSC bit is used to synchronize the internally generated clock (internal SCLi) and an external clock supplied to the SCLi pin. In cases when the CSC bit is set to "1" (clock synchronization enabled), if a falling edge on the SCLi pin is detected while the internal SCLi is high, the internal SCLi goes low, at which time the UiBRG register value is reloaded with and starts counting in the low-level interval. If the internal SCLi changes state from low to high while the SCLi pin is low, counting stops, and when the SCLi pin goes high, counting restarts.
In this way, the UARTi transfer clock is comprised of the logical product of the internal SCLi and SCLi pin signal. The transfer clock works from a half period before the falling edge of the internal SCLi 1st bit to the rising edge of the 9th bit. To use this function, select an internal clock for the transfer clock.
The UiSMR2 register's SWC bit allows to select whether the SCLi pin should be fixed to or freed from low-level output at the falling edge of the 9th clock pulse.
If the UiSMR4 register's SCLHI bit is set to "1" (enabled), SCLi output is turned off (placed in the high impedance state) when a stop condition is detected.
Setting the UiSMR2 register's SWC2 bit = 1 (0 output) makes it possible to forcibly output a low-level signal from the SCLi pin even while sending or receiving data. Clearing the SWC2 bit to "0" (transfer clock) allows the transfer clock to be output from or supplied to the SCLi pin, instead of outputting a low-level signal.
If the UiSMR4 register's SWC9 bit is set to "1" (SCL hold low enabled) when the UiSMR3 register's CKPH bit = 1, the SCLi pin is fixed to low-level output at the falling edge of the clock pulse next to the ninth. Setting the SWC9 bit = 0 (SCL hold low disabled) frees the SCLi pin from low-level output.

### 10.4.5 SDA Output

The data written to the UiTB register bit 7 to bit 0 ($D_7$ to $D_0$) is sequentially output beginning with $D_7$.
The ninth bit ($D_8$) is ACK or NACK.
The initial value of SDAi transmit output can only be set when IICM = 1 ($I^2C$ mode) and the UiMR register's SMD2 to SMD0 bits = '$000_2$' (serial I/O disabled).
The UiSMR3 register's DL2 to DL0 bits allow to add no delays or a delay of 2 to 8 UiBRG count source clock cycles to SDAi output.
Setting the UiSMR2 register's SDHI bit = 1 (SDA output disabled) forcibly places the SDAi pin in the high-impedance state. Do not write to the SDHI bit synchronously with the rising edge of the UARTi transfer clock. This is because the ABT bit may inadvertently be set to "1" (detected).

### 10.4.6 SDA Input

When the IICM2 bit = 0, the 1st to 8th bits ($D_7$ to $D_0$) of received data are stored in the UiRB register bit 7 to bit 0. The 9th bit ($D_8$) is ACK or NACK.
When the IICM2 bit = 1, the 1st to 7th bits ($D_7$ to $D_1$) of received data are stored in the UiRB register bit 6 to bit 0 and the 8th bit ($D_0$) is stored in the UiRB register bit 8. Even when the IICM2 bit = 1, providing the CKPH bit = 1, the same data as when the IICM2 bit = 0 can be read out by reading the UiRB register after the rising edge of the corresponding clock pulse of 9th bit.

### 10.4.7 ACK and NACK

If the STSPSEL bit in the UiSMR4 register is set to "0" (start and stop conditions not generated) and the ACKC bit in the UiSMR4 register is se to "1" (ACK data output), the value of the ACKD bit in the UiSMR4 register is output from the SDAi pin.
If the IICM2 bit = 0, a NACK interrupt request is generated if the SDAi pin remains high at the rising edge of the 9th bit of transmit clock pulse. An ACK interrupt request is generated if the SDAi pin is low at the rising edge of the 9th bit of transmit clock pulse.
If ACKi is selected for the cause of DMA1 request, a DMA transfer can be activated by detection of an acknowledge.

### 10.4.8 Initialization of Transmission/Reception

If a start condition is detected while the STAC bit = 1 (UARTi initialization enabled), the serial I/O operates as described below.

The transmit shift register is initialized, and the content of the UiTB register is transferred to the transmit shift register. In this way, the serial I/O starts sending data synchronously with the next clock pulse applied. However, the UARTi output value does not change state and remains the same as when a start condition was detected until the first bit of data is output synchronously with the input clock.

The receive shift register is initialized, and the serial I/O starts receiving data synchronously with the next clock pulse applied.

The SWC bit is set to “1” (SCL wait output enabled). Consequently, the SCLi pin is pulled low at the falling edge of the ninth clock pulse.

Note that when UARTi transmission/reception is started using this function, the TI does not change state. Note also that when using this function, the selected transfer clock should be an external clock.

## 10.5 Special Mode 2

Multiple slaves can be serially communicated from one master. Transfer clock polarity and phase are selectable. Table 10.14 lists the specifications of Special Mode 2. Table 10.15 lists the registers used in Special Mode 2 and the register values set. Figure 10.25 shows communication control example for Special Mode 2.

**Table 10.14 Special Mode 2 Specifications**

| Item | Specification |
|---|---|
| Transfer data format | • Transfer data length: 8 bits |
| Transfer clock | • Master mode<br>UiMR(i=0 to 2) register's CKDIR bit = "0" (internal clock) : fj/ 2(n+1)<br>fj = $f_{1SIO}$, $f_{2SIO}$, $f_{8SIO}$, $f_{32SIO}$. n: Setting value of UiBRG register $00_{16}$ to $FF_{16}$<br>• Slave mode<br>CKDIR bit = "1" (external clock selected) : Input from CLKi pin |
| Transmit/receive control | Controlled by input/output ports |
| Transmission start condition | • Before transmission can start, the following requirements must be met (Note 1)<br>– The TE bit of UiC1 register= 1 (transmission enabled)<br>– The TI bit of UiC1 register = 0 (data present in UiTB register) |
| Reception start condition | • Before reception can start, the following requirements must be met (Note 1)<br>– The RE bit of UiC1 register= 1 (reception enabled)<br>– The TE bit of UiC1 register= 1 (transmission enabled)<br>– The TI bit of UiC1 register= 0 (data present in the UiTB register) |
| Interrupt request generation timing | • For transmission, one of the following conditions can be selected<br>– The UiIRS bit of UiC1 register = 0 (transmit buffer empty): when transferring data from the UiTB register to the UARTi transmit register (at start of transmission)<br>– The UiIRS bit =1 (transfer completed): when the serial I/O finished sending data from the UARTi transmit register<br>• For reception<br>When transferring data from the UARTi receive register to the UiRB register (at completion of reception) |
| Error detection | • Overrun error (Note 2)<br>This error occurs if the serial I/O started receiving the next data before reading the UiRB register and received the 7th bit of the next data |
| Select function | • Clock phase setting<br>Selectable from four combinations of transfer clock polarities and phases |

Note 1: When an external clock is selected, the conditions must be met while if the UiC0 register's CKPOL bit = "0" (transmit data output at the falling edge and the receive data taken in at the rising edge of the transfer clock), the external clock is in the high state; if the UiC0 register's CKPOL bit = "1" (transmit data output at the rising edge and the receive data taken in at the falling edge of the transfer clock), the external clock is in the low state.

Note 2: If an overrun error occurs, the value of UiRB register will be indeterminate. The IR bit of SiRIC register does not change.

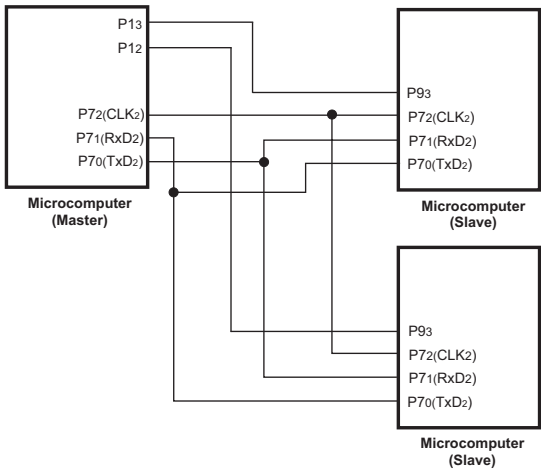



**Figure 10.25 Serial Bus Communication Control Example (UART2)**

**Table 10.15 Registers to Be Used and Settings in Special Mode 2**

| Register | Bit | Function |
|---|---|---|
| UiTB(Note3) | 0 to 7 | Set transmission data |
| UiRB(Note3) | 0 to 7 | Reception data can be read |
| | OER | Overrun error flag |
| UiBRG | 0 to 7 | Set a transfer rate |
| UiMR(Note3) | SMD2 to SMD0 | Set to '001$_2$' |
| | CKDIR | Set this bit to "0" for master mode or "1" for slave mode |
| | IOPOL | Set to "0" |
| UiC0 | CLK1, CLK0 | Select the count source for the UiBRG register |
| | CRS | Invalid because CRD = 1 |
| | TXEPT | Transmit register empty flag |
| | CRD | Set to "1" |
| | NCH | Select TxDi pin output format(Note 2) |
| | CKPOL | Clock phases can be set in combination with the UiSMR3 register's CKPH bit |
| | UFORM | Set to "0" |
| UiC1 | TE | Set this bit to "1" to enable transmission |
| | TI | Transmit buffer empty flag |
| | RE | Set this bit to "1" to enable reception |
| | RI | Reception complete flag |
| | U2IRS (Note 1) | Select UART2 transmit interrupt cause |
| | U2RRM(Note 1), U2LCH, UiERE | Set to "0" |
| UiSMR | 0 to 7 | Set to "0" |
| UiSMR2 | 0 to 7 | Set to "0" |
| UiSMR3 | CKPH | Clock phases can be set in combination with the UiC0 register's CKPOL bit |
| | NODC | Set to "0" |
| | 0, 2, 4 to 7 | Set to "0" |
| UiSMR4 | 0 to 7 | Set to "0" |
| UCON | U0IRS, U1IRS | Select UART0 and UART1 transmit interrupt cause |
| | U0RRM, U1RRM | Set to "0" |
| | CLKMD0 | Invalid because CLKMD1 = 0 |
| | CLKMD1, RCSP, 7 | Set to "0" |

Note 1: Set the U0C0 and U1C1 register bit 4 and bit 5 to "0". The U0IRS, U1IRS, U0RRM and U1RRM bits are in the UCON register.
Note 2: TxD2 pin is N channel open-drain output. Set the U2C0 register's NCH bit to "0".
Note 3: Not all register bits are described above. Set those bits to "0" when writing to the registers in Special Mode 2.
i = 0 to 2

### 10.5.1 Clock Phase Setting Function

One of four combinations of transfer clock phases and polarities can be selected using the UiSMR3 register's CKPH bit and the UiC0 register's CKPOL bit.

Make sure the transfer clock polarity and phase are the same for the master and salves to be communicated.

- Master (Internal Clock)
  Figure 10.26 shows the transmission and reception timing in master (internal clock).
- Slave (External Clock)
  Figure 10.27 shows the transmission and reception timing (CKPH=0) in slave (external clock) while Figure 10.28 shows the transmission and reception timing (CKPH=1) in slave (external clock).

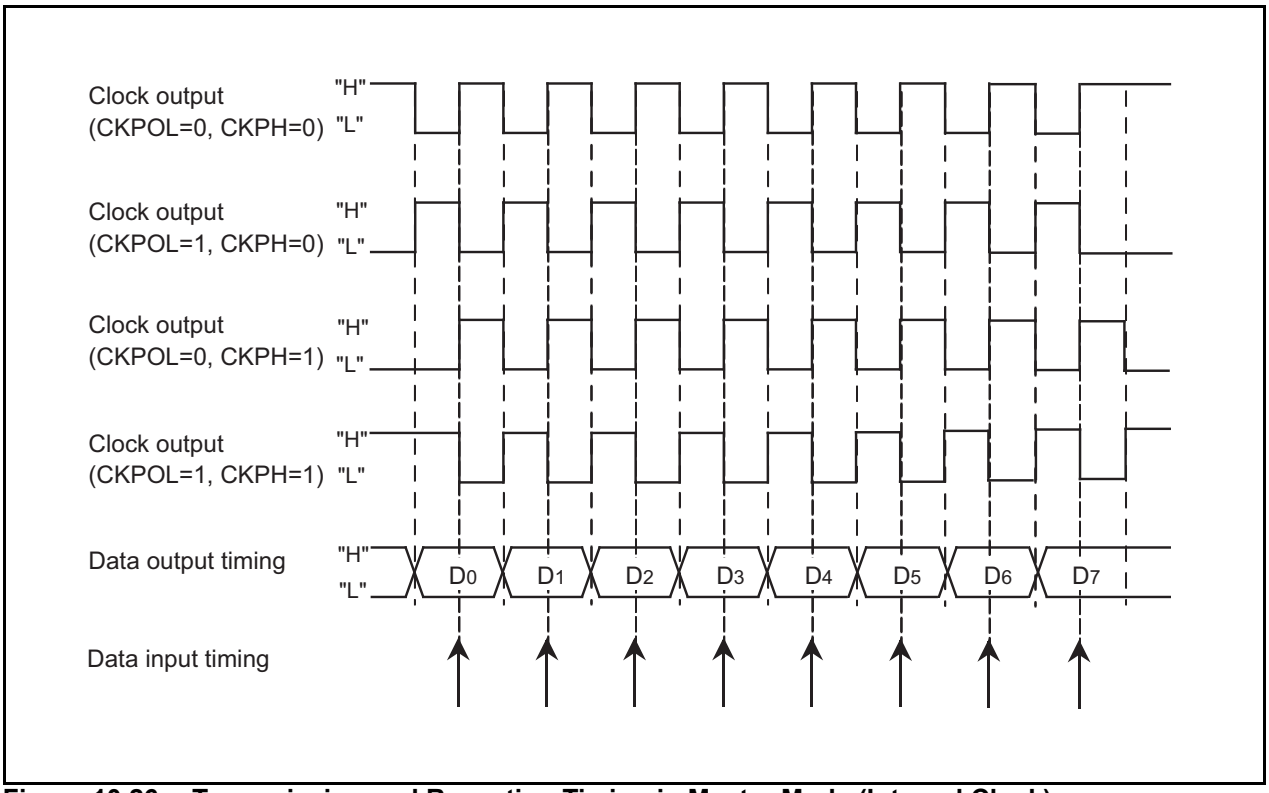
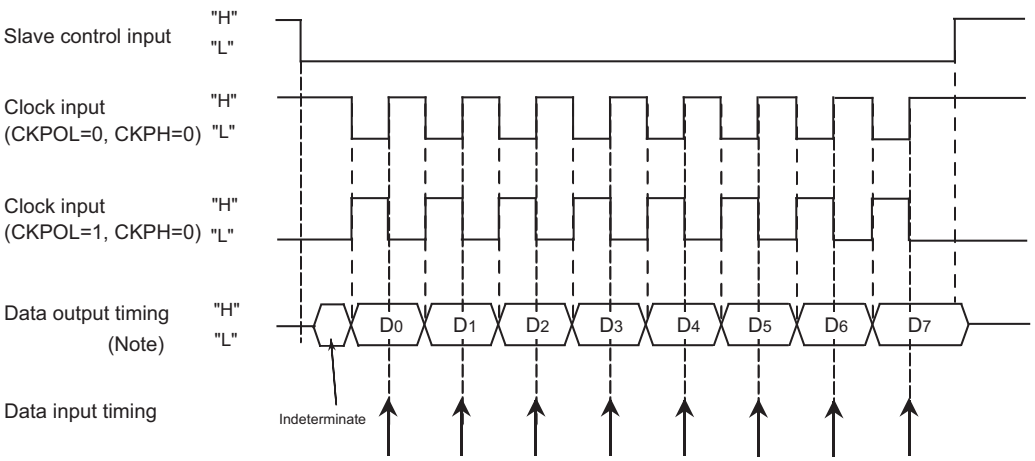
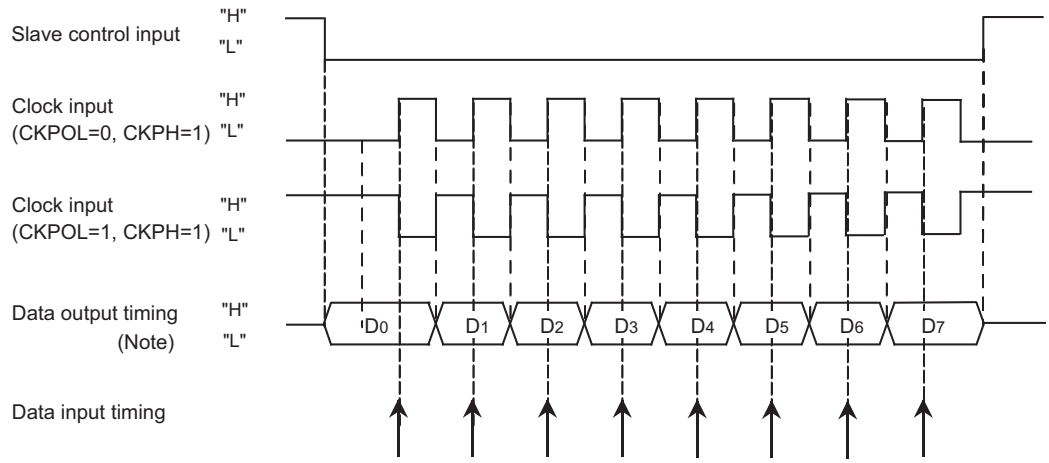



**Figure 10.26 Transmission and Reception Timing in Master Mode (Internal Clock)**




Note :UART2 output is an N-channel open drain and must be pulled-up externally.

**Figure 10.27 Transmission and Reception Timing (CKPH=0) in Slave Mode (External Clock)**




Note :UART2 output is an N-channel open drain and must be pulled-up externally.

**Figure 10.28 Transmission and Reception Timing (CKPH=1) in Slave Mode (External Clock)**

## 10.6 Special Mode 3 (IE mode)

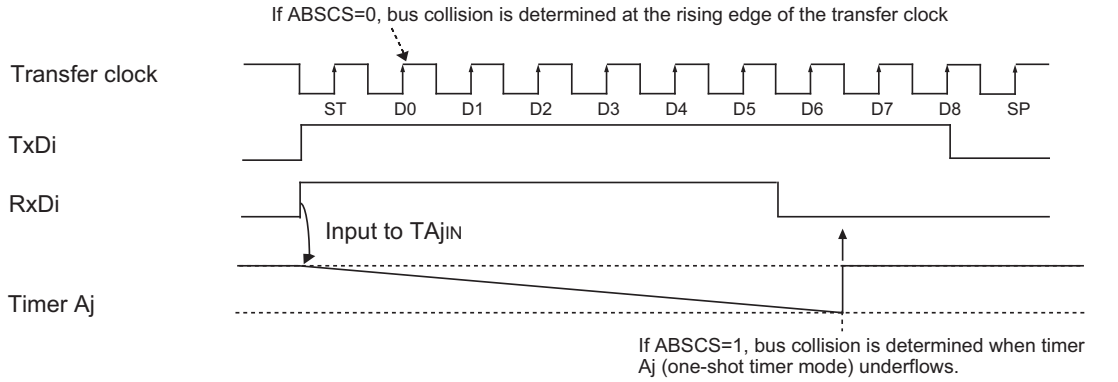
In this mode, one bit of IEBus is approximated with one byte of UART mode waveform.
Table 10.16 lists the registers used in IE mode and the register values set. Figure 10.29 shows the functions of bus collision detect function related bits.
If the TxDi pin (i = 0 to 2) output level and RxDi pin input level do not match, a UARTi bus collision detect interrupt request is generated.
Use the IFSR2A register's IFSR26 and IFSR27 bits to enable the UART0/UART1 bus collision detect function.

**Table 10.16 Registers to Be Used and Settings in IE Mode**

| Register | Bit | Function |
|---|---|---|
| UiTB | 0 to 8 | Set transmission data |
| UiRB(Note3) | 0 to 8 | Reception data can be read |
| | OER,FER,PER,SUM | Error flag |
| UiBRG | 0 to 7 | Set a transfer rate |
| UiMR | SMD2 to SMD0 | Set to '$110_2$' |
| | CKDIR | Select the internal clock or external clock |
| | STPS | Set to "0" |
| | PRY | Invalid because PRYE=0 |
| | PRYE | Set to "0" |
| | IOPOL | Select the TxD/RxD input/output polarity |
| UiC0 | CLK1, CLK0 | Select the count source for the UiBRG register |
| | CRS | Invalid because CRD=1 |
| | TXEPT | Transmit register empty flag |
| | CRD | Set to "1" |
| | NCH | Select TxDi pin output mode (Note 2) |
| | CKPOL | Set to "0" |
| | UFORM | Set to "0" |
| UiC1 | TE | Set this bit to "1" to enable transmission |
| | TI | Transmit buffer empty flag |
| | RE | Set this bit to "1" to enable reception |
| | RI | Reception complete flag |
| | U2IRS (Note 1) | Select the source of UART2 transmit interrupt |
| | UiRRM (Note 1), UiLCH, UiERE | Set to "0" |
| UiSMR | 0 to 3, 7 | Set to "0" |
| | ABSCS | Select the sampling timing at which to detect a bus collision |
| | ACSE | Set this bit to "1" to use the auto clear function of transmit enable bit |
| | SSS | Select the transmit start condition |
| UiSMR2 | 0 to 7 | Set to "0" |
| UiSMR3 | 0 to 7 | Set to "0" |
| UiSMR4 | 0 to 7 | Set to "0" |
| IFSR2A | IFSR26, IFSR27 | Set to "1" |
| UCON | U0IRS, U1IRS | Select the source of UART0/UART1 transmit interrupt |
| | U0RRM, U1RRM | Set to "0" |
| | CLKMD0 | Invalid because CLKMD1 = 0 |
| | CLKMD1,RCSP,7 | Set to "0" |

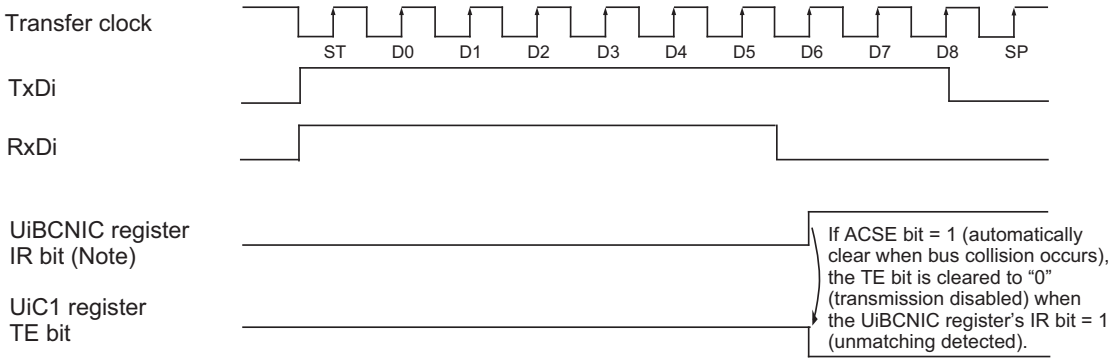
Note 1: Set the U0C0 and U1C1 registers bit 4 and bit 5 to "0". The U0IRS, U1IRS, U0RRM and U1RRM bits are in the UCON register.
Note 2: TxD2 pin is N channel open-drain output. Set the U2C0 register's NCH bit to "0".
Note 3: Not all register bits are described above. Set those bits to "0" when writing to the registers in IEmode.
i= 0 to 2

**(1) UiSMR register ABSCS bit (bus collision detect sampling clock select)** (i=0 to 2)




Timer Aj: timer A3 when UART0; timer A4 when UART1; timer A0 when UART2

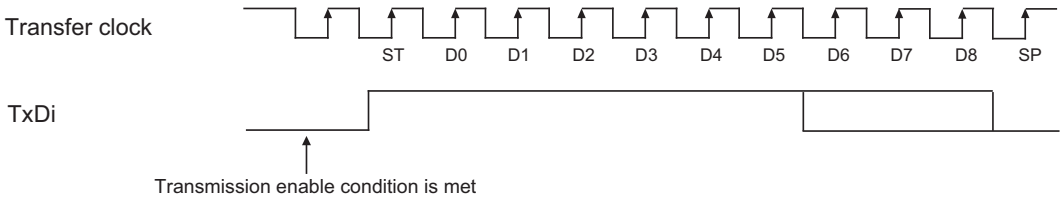
**(2) UiSMR register ACSE bit (auto clear of transmit enable bit)**



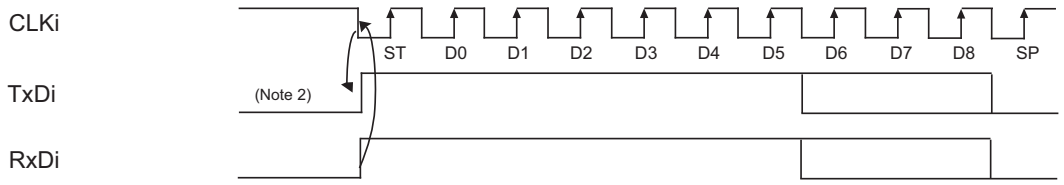

Note: BCNIC register when UART2.

**(3) UiSMR register SSS bit (Transmit start condition select)**

If SSS bit = 0, the serial I/O starts sending data one transfer clock cycle after the transmission enable condition is met.




If SSS bit = 1, the serial I/O starts sending data at the rising edge (Note 1) of RxDi




Note 1: The falling edge of RxDi when IOPOL=0; the rising edge of RxDi when IOPOL =1.
Note 2: The transmit condition must be met before the falling edge (Note 1) of RxD.

This diagram applies to the case where IOPOL=1 (reversed).

**Figure 10.29 Bus Collision Detect Function-Related Bits**

## 10.7 Special Mode 4 (SIM Mode) (UART2)

Based on UART mode, this is an SIM interface compatible mode. Direct and inverse formats can be implemented, and this mode allows to output a low from the TxD2 pin when a parity error is detected.
Tables 10.17 lists the specifications of SIM mode. Table 10.18 lists the registers used in the SIM mode and the register values set.

**Table 10.17 SIM Mode Specifications**

| Item | Specification |
|---|---|
| Transfer data format | • Direct format<br>• Inverse format |
| Transfer clock | • U2MR register's CKDIR bit = "0" (internal clock) : fi/ 16(n+1)<br>fi = $f_{1SIO}$, $f_{2SIO}$, $f_{8SIO}$, $f_{32SIO}$. n: Setting value of U2BRG register $00_{16}$ to $FF_{16}$<br>• CKDIR bit = "1" (external clock) : $f_{EXT}$/16(n+1)<br>$f_{EXT}$: Input from $CLK_2$ pin. n: Setting value of U2BRG register $00_{16}$ to $FF_{16}$ |
| Transmission start condition | • Before transmission can start, the following requirements must be met<br>– The TE bit of U2C1 register= 1 (transmission enabled)<br>– The TI bit of U2C1 register = 0 (data present in U2TB register) |
| Reception start condition | • Before reception can start, the following requirements must be met<br>– The RE bit of U2C1 register= 1 (reception enabled)<br>– Start bit detection |
| Interrupt request generation timing (Note 2) | • For transmission<br>When the serial I/O finished sending data from the U2TB transfer register (U2IRS bit =1)<br>• For reception<br>When transferring data from the UART2 receive register to the U2RB register (at completion of reception) |
| Error detection | • Overrun error (Note 1)<br>This error occurs if the serial I/O started receiving the next data before reading the U2RB register and received the bit one before the last stop bit of the next data<br>• Framing error<br>This error occurs when the number of stop bits set is not detected<br>• Parity error<br>During reception, if a parity error is detected, parity error signal is output from the $TxD_2$ pin.<br>During transmission, a parity error is detected by the level of input to the $RxD_2$ pin when a transmission interrupt occurs<br>• Error sum flag<br>This flag is set (= 1) when any of the overrun, framing, and parity errors is encountered |

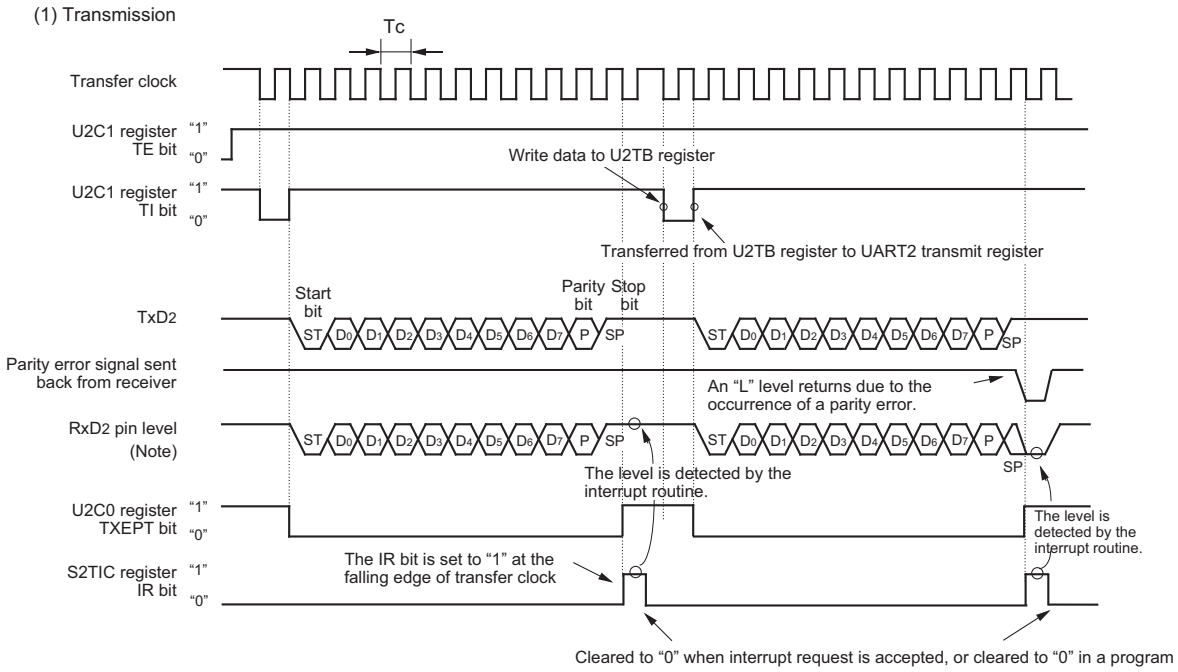
Note 1: If an overrun error occurs, the value of U2RB register will be indeterminate. The IR bit of S2RIC register does not change.

Note 2: A transmit interrupt request is generated by setting the U2C1 register U2IRS bit to "1" (transmission complete) and U2ERE bit to "1" (error signal output) after reset. Therefore, when using SIM mode, be sure to clear the IR bit to "0" (no interrupt request) after setting these bits.

**Table 10.18 Registers to Be Used and Settings in SIM Mode**

| Register | Bit | Function |
|---|---|---|
| U2TB(Note) | 0 to 7 | Set transmission data |
| U2RB(Note) | 0 to 7 | Reception data can be read |
| | OER,FER,PER,SUM | Error flag |
| U2BRG | 0 to 7 | Set a transfer rate |
| U2MR | SMD2 to SMD0 | Set to '$101_2$' |
| | CKDIR | Select the internal clock or external clock |
| | STPS | Set to "0" |
| | PRY | Set this bit to "1" for direct format or "0" for inverse format |
| | PRYE | Set to "1" |
| | IOPOL | Set to "0" |
| U2C0 | CLK1, CLK0 | Select the count source for the U2BRG register |
| | CRS | Invalid because CRD=1 |
| | TXEPT | Transmit register empty flag |
| | CRD | Set to "1" |
| | NCH | Set to "0" |
| | CKPOL | Set to "0" |
| | UFORM | Set this bit to "0" for direct format or "1" for inverse format |
| U2C1 | TE | Set this bit to "1" to enable transmission |
| | TI | Transmit buffer empty flag |
| | RE | Set this bit to "1" to enable reception |
| | RI | Reception complete flag |
| | U2IRS | Set to "1" |
| | U2RRM | Set to "0" |
| | U2LCH | Set this bit to "0" for direct format or "1" for inverse format |
| | U2ERE | Set to "1" |
| U2SMR(Note) | 0 to 3 | Set to "0" |
| U2SMR2 | 0 to 7 | Set to "0" |
| U2SMR3 | 0 to 7 | Set to "0" |
| U2SMR4 | 0 to 7 | Set to "0" |

Note: Not all register bits are described above. Set those bits to "0" when writing to the registers in SIM mode.

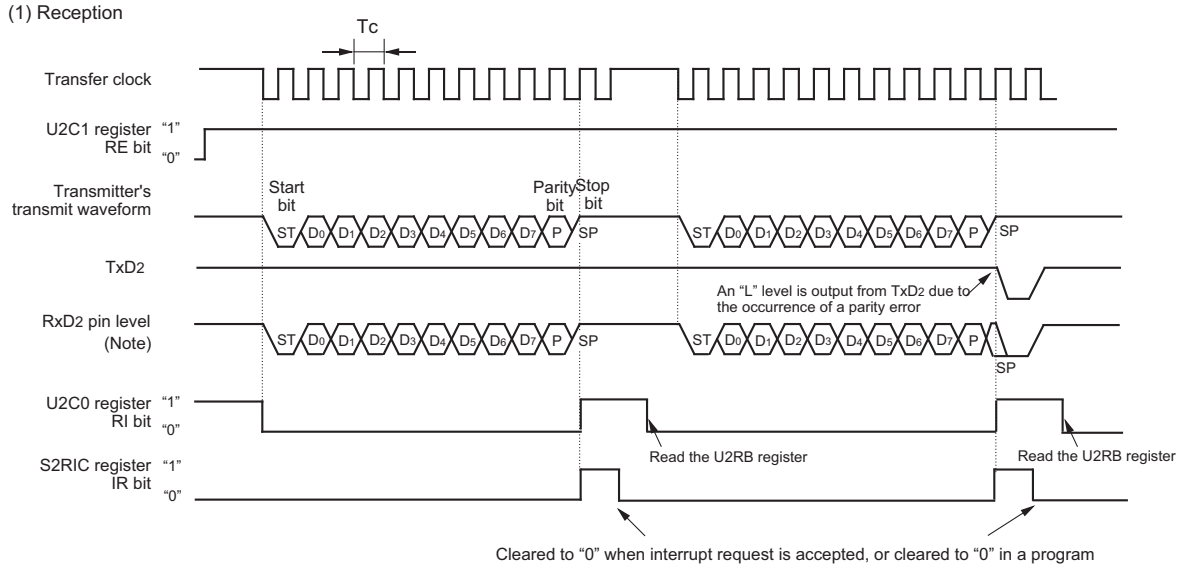



The above timing diagram applies to the case where data is transferred in the direct format.

- U2MR register STPS bit = 0 (1 stop bit)
- U2MR register PRY bit = 1 (even)
- U2C0 register UFORM bit = 0 (LSB first)
- U2C1 register U2LCH bit = 0 (no reverse)
- U2C1 register U2IRSCH bit = 1 (transmit is completed)

Tc = 16 (n + 1) / fi or 16 (n + 1) / fEXT
fi : frequency of U2BRG count source (f1SIO, f2SIO, f8SIO, f32SIO)
fEXT : frequency of U2BRG count source (external clock)
n : value set to U2BRG

Note : Because TxD2 and RxD2 are connected, this is composite waveform consisting of the TxD2 output and the parity error signal sent back from receiver.




The above timing diagram applies to the case where data is received in direct format.

- U2MR register STPS bit = 0 (1 stop bit)
- U2MR register PRY bit = 1 (even)
- U2C0 register UFORM bit = 0 (LSB first)
- U2C1 register U2LCH bit = 0 (no reverse)
- U2C1 register U2IRSCH bit = 1 (transmit is completed)

Tc = 16 (n + 1) / fi or 16 (n + 1) / fEXT
fi : frequency of U2BRG count source (f1SIO, f2SIO, f8SIO, f32SIO)
fEXT : frequency of U2BRG count source (external clock)
n : value set to U2BRG

Note : Because TxD2 and RxD2 are connected, this is composite waveform consisting of the transmitter's transmit waveform and the parity error signal received.

**Figure 10.30 Transmit and Receive Timing in SIM Mode**

Figure 10.31 shows the example of connecting the SIM interface. Connect TxD2 and RxD2 and apply pull-up.

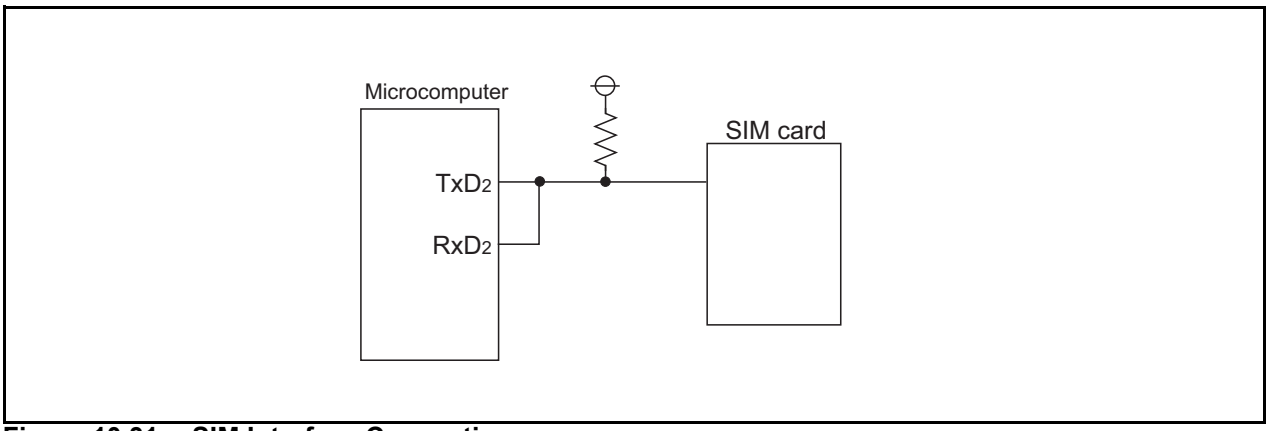



**Figure 10.31 SIM Interface Connection**

### 10.7.1 Parity Error Signal Output

The parity error signal is enabled by setting the U2C1 register's U2ERE bit to "1".

- When receiving

  The parity error signal is output when a parity error is detected while receiving data. This is achieved by pulling the TxD2 output low with the timing shown in Figure 10.32. If the U2RB register is read while outputting a parity error signal, the PER bit is cleared to "0" and at the same time the TxD2 output is returned high.

- When transmitting

  A transmission-finished interrupt request is generated at the falling edge of the transfer clock pulse that immediately follows the stop bit. Therefore, whether a parity signal has been returned can be determined by reading the port that shares the RxD2 pin in a transmission-finished interrupt service routine.

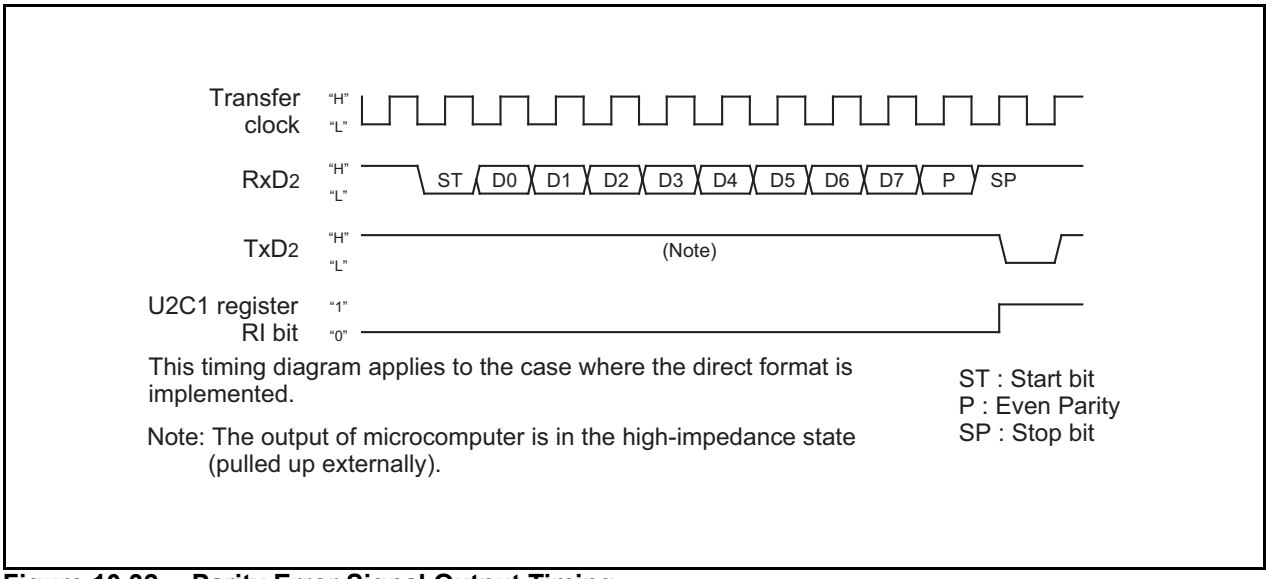



**Figure 10.32 Parity Error Signal Output Timing**

### 10.7.2 Format

• Direct Format
Set the U2MR register's PRY bit to “1”, U2C0 register's UFORM bit to “0” and U2C1 register's U2LCH bit to “0”.

• Inverse Format
Set the PRY bit to “0”, UFORM bit to “1” and U2LCH bit to “1”. Figure 10.33 shows the SIM interface format.

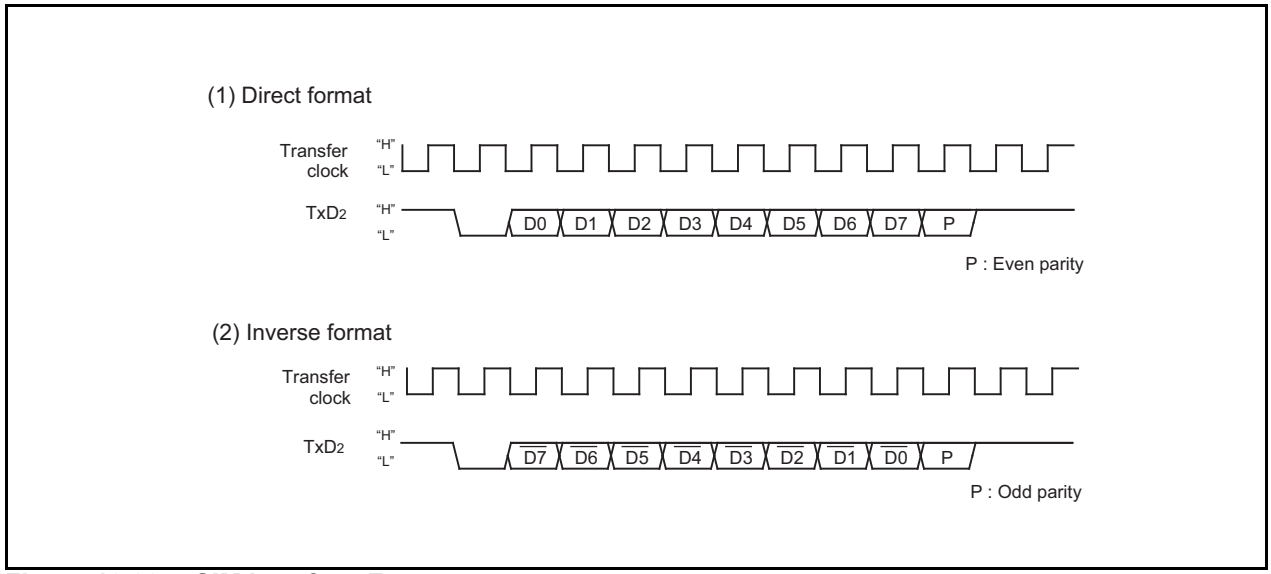



**Figure 10.33 SIM Interface Format**

## 10.8 SI/O3 and SI/O4

SI/O3 and SI/O4 are exclusive clock-synchronous serial I/Os.
Figure 10.34 shows the block diagram of SI/O3 and SI/O4, and Figure 10.35 shows the SI/O3 and SI/O4- related registers.
Table 10.19 shows the specifications of SI/O3 and SI/O4.

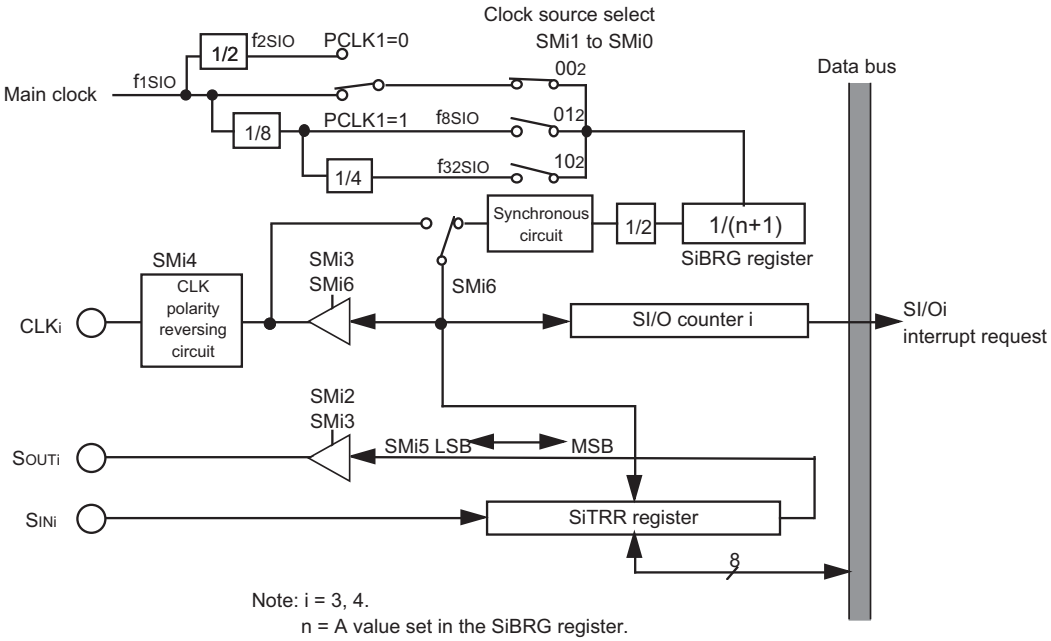



**Figure 10.34 SI/O3 and SI/O4 Block Diagram**

S I/Oi control register (i = 3, 4) (Note 1)

b7 b6 b5 b4 b3 b2 b1 b0

| Symbol | Address | After reset |
|---|---|---|
| S3C | $036216$ | $01000000_{16}$ |
| S4C | $036616$ | $01000000_{16}$ |

| Bit symbol | Bit name | Description | RW |
|---|---|---|---|
| SMi0 | Internal synchronous clock select bit | b1 b0<br>0 0 : Selecting $f_{1SIO}$ or $f_{2SIO}$<br>0 1 : Selecting $f_{8SIO}$<br>1 0 : Selecting $f_{32SIO}$<br>1 1 : Must not be set. | RW |
| SMi1 | | | RW |
| SMi2 | SOUTi output disable bit (Note 4) | 0 : SOUTi output<br>1 : SOUTi output disable(high impedance) | RW |
| SMi3 | S I/Oi port select bit | 0 : Input/output port<br>1 : SOUTi output, CLKi function | RW |
| SMi4 | CLK polarity select bit | 0 : Transmit data is output at falling edge of transfer clock and receive data is input at rising edge<br>1 : Transmit data is output at rising edge of transfer clock and receive data is input at falling edge | RW |
| SMi5 | Transfer direction select bit | 0 : LSB first<br>1 : MSB first | RW |
| SMi6 | Synchronous clock select bit | 0 : External clock (Note 2)<br>1 : Internal clock (Note 3) | RW |
| SMi7 | SOUTi initial value set bit | Effective when SMi3 = 0<br>0 : "L" output<br>1 : "H" output | RW |

Note 1: Make sure this register is written to by the next instruction after setting the PRCR register's PRC2 bit to "1" (write enable).
Note 2: Set the SMi3 bit to "1" and the corresponding port direction bit to "0" (input mode).
Note 3: Set the SMi3 bit to "1" (SOUTi output, CLKi function).
Note 4: When the SMi2 bit is set to "1", the target pin goes to a high-impedance state regardless of which function of the pin is being used.

SI/Oi bit rate generator (i = 3, 4) (Notes 1, 2)

b7 b0

| Symbol | Address | After reset |
|---|---|---|
| S3BRG | $036316$ | Indeterminate |
| S4BRG | $036716$ | Indeterminate |

| Description | Setting range | RW |
|---|---|---|
| Assuming that set value = n, BRGi divides the count source by n + 1 | $00_{16}$ to $FF_{16}$ | WO |

Note 1: Write to this register while serial I/O is neither transmitting nor receiving.
Note 2: Use MOV instruction to write to this register.

SI/Oi transmit/receive register (i = 3, 4) (Note 1, 2)

b7 b0

| Symbol | Address | After reset |
|---|---|---|
| S3TRR | $036016$ | Indeterminate |
| S4TRR | $036416$ | Indeterminate |

| Description | RW |
|---|---|
| Transmission/reception starts by writing transmit data to this register. After transmission/reception finishes, reception data can be read by reading this register. | RW |

Note 1: Write to this register while serial I/O is neither transmitting nor receiving.
Note 2: To receive data, set the corresponding port direction bit for SINi to "0" (input mode).

**Figure 10.35 S3C and S4C Registers, S3BRG and S4BRG Registers, and S3TRR and S4TRR Registers**

**Table 10.19 SI/O3 and SI/O4 Specifications**

| Item | Specification |
|---|---|
| Transfer data format | • Transfer data length: 8 bits |
| Transfer clock | • SiC (i=3, 4) register's SMi6 bit = "1" (internal clock) : fj/ 2(n+1)<br>fj = f1SIO, f8SIO, f32SIO. n=Setting value of SiBRG register $00_{16}$ to $FF_{16}$.<br>• SMi6 bit = "0" (external clock) : Input from CLKi pin (Note 1) |
| Transmission/reception start condition | • Before transmission/reception can start, the following requirements must be met<br>Write transmit data to the SiTRR register (Notes 2, 3) |
| Interrupt request generation timing | • When SiC register's SMi4 bit = 0<br>The rising edge of the last transfer clock pulse (Note 4)<br>• When SMi4 = 1<br>The falling edge of the last transfer clock pulse (Note 4) |
| CLKi pin function | I/O port, transfer clock input, transfer clock output |
| SOUTi pin function | I/O port, transmit data output, high-impedance |
| SINi pin function | I/O port, receive data input |
| Select function | • LSB first or MSB first selection<br>Whether to start sending/receiving data beginning with bit 0 or beginning with bit 7 can be selected<br>• Function for setting an SOUTi initial value set function<br>When the SiC register's SMi6 bit = 0 (external clock), the SOUTi pin output level while not transmitting can be selected<br>• CLK polarity selection<br>Whether transmit data is output/input timing at the rising edge or falling edge of transfer clock can be selected. |

Note 1: To set the SiC register's SMi6 bit to "0" (external clock), follow the procedure described below.
• If the SiC register's SMi4 bit = 0, write transmit data to the SiTRR register while input on the CLKi pin is high. The same applies when rewriting the SiC register's SMi7 bit.
• If the SMi4 bit = 1, write transmit data to the SiTRR register while input on the CLKi pin is low. The same applies when rewriting the SMi7 bit.
• Because shift operation continues as long as the transfer clock is supplied to the SI/Oi circuit, stop the transfer clock after supplying eight pulses. If the SMi6 bit = 1 (internal clock), the transfer clock automatically stops.

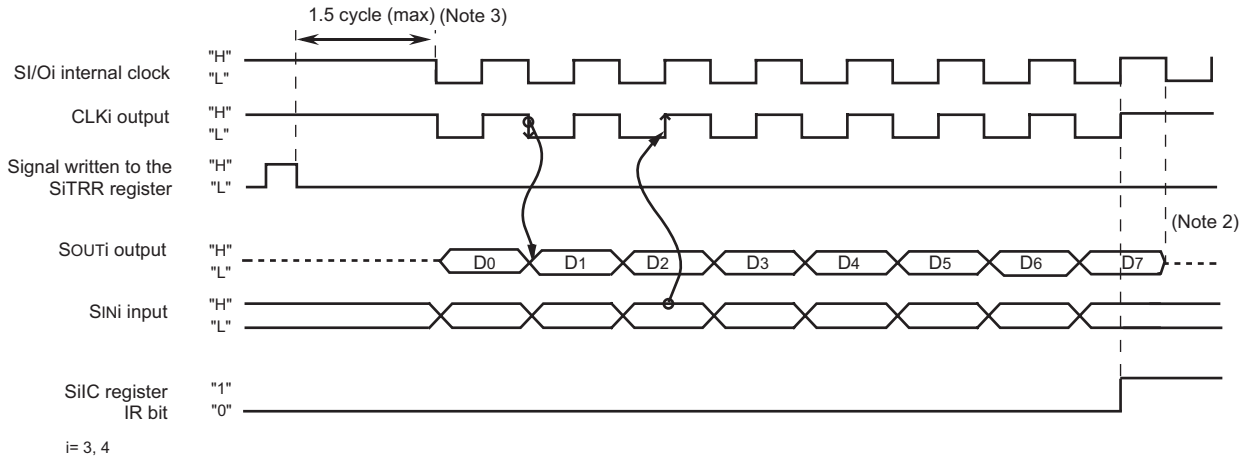
Note 2: Unlike UART0 to UART2, SI/Oi (i = 3 to 4) is not separated between the transfer register and buffer. Therefore, do not write the next transmit data to the SiTRR register during transmission.

Note 3: When the SiC register's SMi6 bit = 1 (internal clock), SOUTi retains the last data for a 1/2 transfer clock period after completion of transfer and, thereafter, goes to a high-impedance state. However, if transmit data is written to the SiTRR register during this period, SOUTi immediately goes to a high-impedance state, with the data hold time thereby reduced.

Note 4: When the SiC register's SMi6 bit = 1 (internal clock), the transfer clock stops in the high state if the SMi4 bit = 0, or stops in the low state if the SMi4 bit = 1.

### 10.8.1 SI/Oi Operation Timing

Figure 10.36 shows the SI/Oi operation timing

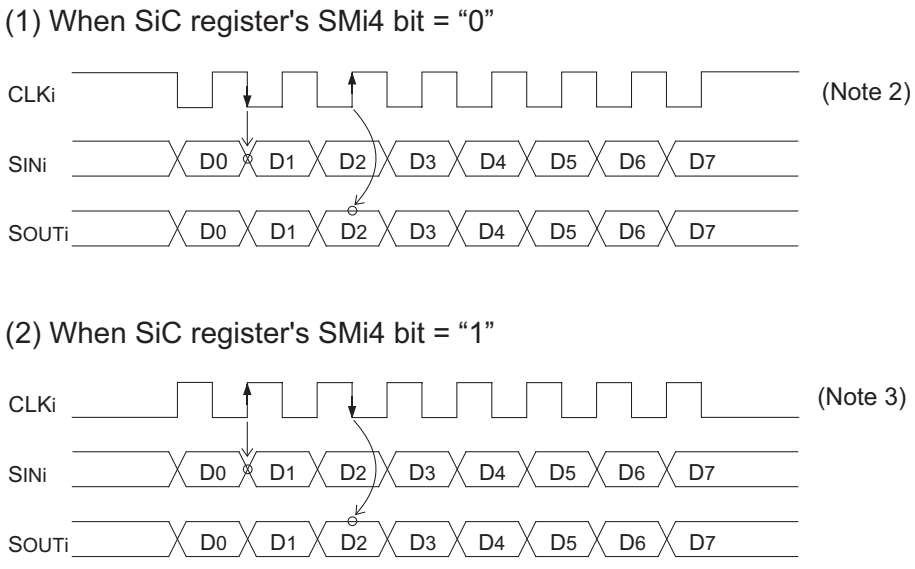



Note 1: This diagram applies to the case where the SiC register bits are set as follows:
SMi2=0 (SOUTi output), SMi3=1 (SOUTi output, CLKi function), SMi4=0 (transmit data output at the falling edge and receive data input at the rising edge of the transfer clock), SMi5=0 (LSB first) and SMi6=1 (internal clock)
Note 2: When the SMi6 bit = 1 (internal clock), the SOUTi pin is placed in the high-impedance state after the transfer finishes.
Note 3: If the SMi6 bit=0 (internal clock), the serial I/O starts sending or receiving data a maximum of 1.5 transfer clock cycles after writing to the SiTRR register.

**Figure 10.36 SI/Oi Operation Timing**

### 10.8.2 CLK Polarity Selection

The SiC register's SMi4 bit allows selection of the polarity of the transfer clock. Figure 10.37 shows the polarity of the transfer clock.

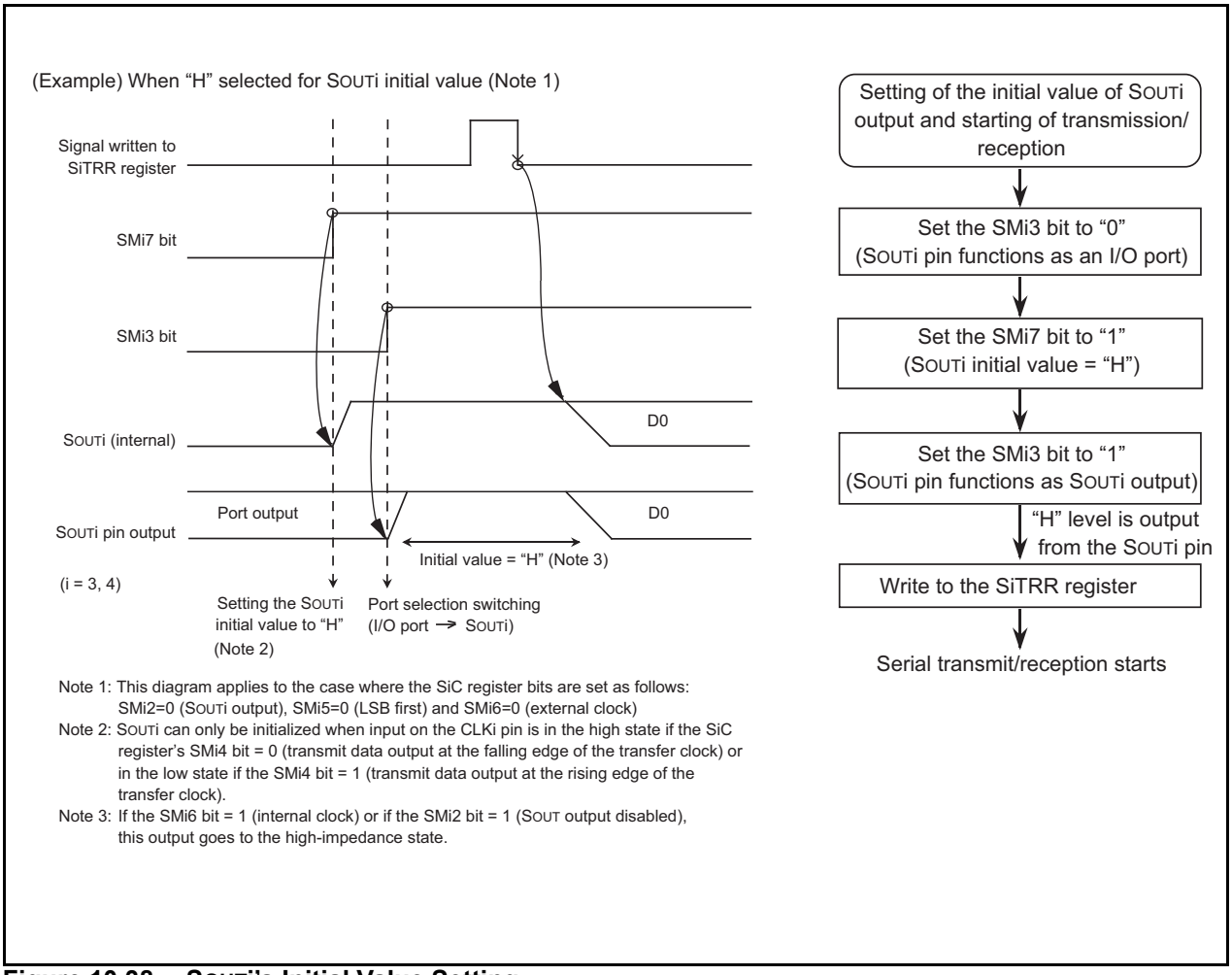



Note 1: This diagram applies to the case where the SiC register bits are set as follows:
SMi5=0 (LSB first) and SMi6=1 (internal clock)
Note 2: When the SMi6 bit=1 (internal clock), a high level is output from the CLKi pin if not transferring data.
Note 3: When the SMi6 bit=1 (internal clock), a low level is output from the CLKi pin if not transferring data.

**Figure 10.37 Polarity of Transfer Clock**

### 10.8.3 Functions for Setting an SOUTi Initial Value

If the SiC register's SMi6 bit = 0 (external clock), the SOUTi pin output can be fixed high or low when not transferring. Figure 10.38 shows the timing chart for setting an SOUTi initial value and how to set it.




Note 1: This diagram applies to the case where the SiC register bits are set as follows: SMi2=0 (SOUTi output), SMi5=0 (LSB first) and SMi6=0 (external clock)
Note 2: SOUTi can only be initialized when input on the CLKi pin is in the high state if the SiC register's SMi4 bit = 0 (transmit data output at the falling edge of the transfer clock) or in the low state if the SMi4 bit = 1 (transmit data output at the rising edge of the transfer clock).
Note 3: If the SMi6 bit = 1 (internal clock) or if the SMi2 bit = 1 (SOUT output disabled), this output goes to the high-impedance state.

**Figure 10.38 SOUTi's Initial Value Setting**

# 11. Multi-master I2C-BUS Interface

The multi-master I2C-BUS interface have each dedicated circuit and operate independently.
The multi-master I2C-BUS interface is a serial communications circuit, conforming to the Philips I2C-BUS data transfer format. This interface i, offering both arbitration lost detection and a synchronous functions, is useful for the multi-master serial communications.
Table 11.1 shows multi-master I2C-BUS interface functions.
This multi-master I2C-BUS interface consists of I2C address register, I2C data shift register, I2C clock control register, I2C control register, I2C status register, I2C transmit buffer register and the other control circuits.

**Table 11.1 Clock Generation Circuit Specifications**

| Item | Function |
|---|---|
| Format | In conformity with Philips I2C-BUS standard:<br>10-bit addressing format<br>7-bit addressing format<br>High-speed clock mode<br>Standard clock mode |
| Communication mode | In conformity with Philips I2C-BUS standard:<br>Master transmission<br>Master reception<br>Slave transmission<br>Slave reception |
| SCL clock frequency | 16.1 kHz to 400 kHz (BCLK = 16 MHz) |
| Power supply voltage on bus line | (SCL3/SDA3) : Vcc1 |

Note. Our company doesn't assume the responsibility of the patent of the third party who originates in the use of the function to control the connection of I2C-BUS interface and ports (SCL3, SDA3) and other infringements of right.

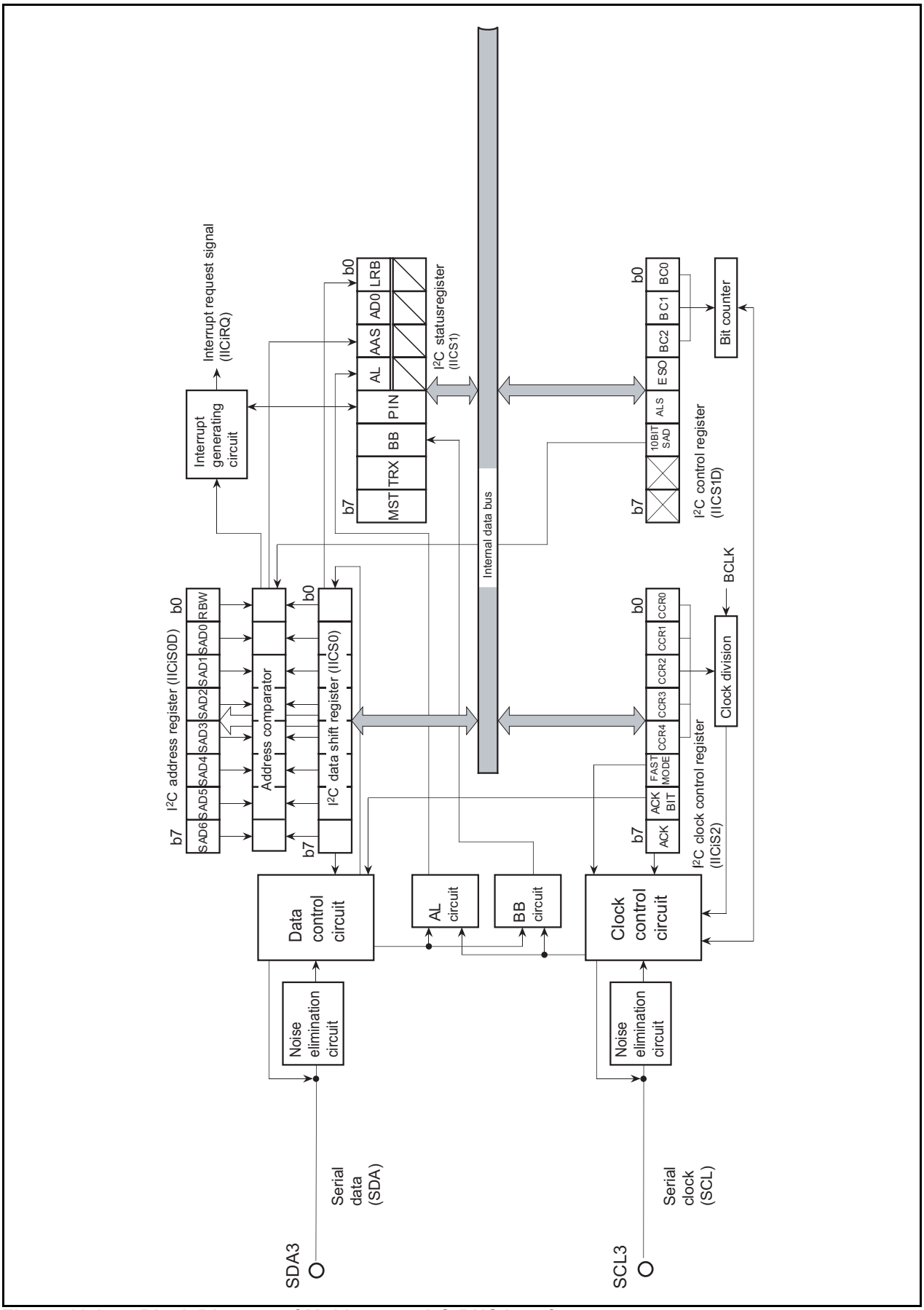



**Figure 11.1 Block Diagram of Multi-master I2C-BUS interface**

### (1) Reserved register

Reserved register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | | 0 | 0 |

Symbol: RSVREG02E5
Address: 02E5$_{16}$
When reset: 00?00000$_2$

| Bit Symbol | Bit name | Function | RW |
|---|---|---|---|
| Reserved bits | | Must always be set to “0” | WO |
| RSVREG02E52 | Multi-master I2C-BUS interface enable bit | 0 = Non active<br>1 = Active | RW |
| Reserved bits | | Must always be set to “0” | WO |

**Figure 11.2 Reserved register**

### (2) I2C data shift register, I2C transmit buffer register

The I2C data shift register is an 8-bit shift register to store receive data and write transmit data.
When transmit data is written into this register, it is transferred to the outside from bit 7 in synchronization with the SCL clock, and each time one-bit data is output, the data of this register are shifted one bit to the left. When data is received, it is input to this register from bit 0 in synchronization with the SCL clock, and each time one-bit data is input, the data of this register are shifted one bit to the left.
The I2C data shift register is in a write enable status only when the ESO bit of the I2C control register is "1." The bit counter is reset by a write instruction to the I2C data shift register. When both the ESO bit and the MST bit of the I2C status register are "1," the SCL is output by a write instruction to the I2C data shift register. Reading data from the I2C data shift register is always enabled regardless of the ESO bit value.
The I2C transmit buffer register is a register to store transmit data (slave address) to the I2C data shift register before RESTART condition generation. That is, in master, transmit data written to the I2C transmit buffer register is written to the I2C data shift register simultaneously. However, the SCL is not output. The I2C transmit buffer register can be written only when the ESO bit is "1," reading data from the I2C transmit buffer register is disabled regardless of the ESO bit value.

**Notes 1:** To write data into the I2C data shift register or the I2C transmit buffer register after the MST bit value changes from "1" to "0" (slave mode), keep an interval of 20 BCLK or more.
**2:** To generate START/RESTART condition after the I2C data shift register or the I2C transmit buffer register is written, keep an interval of 4 BCLK or more.

I2C data shift register

| Symbol | Address | When reset |
|---|---|---|
| IIC0S0 | 02E016 | Indeterminate |

b7 b6 b5 b4 b3 b2 b1 b0

| Bit Symbol | Bit name | Function | RW |
|---|---|---|---|
| D0 | Data shift register | This is an 8-bit shift register to store receive data and write transmit data. | RW |
| D1 | | | |
| D2 | | | |
| D3 | | | |
| D4 | | | |
| D5 | | | |
| D6 | | | |
| D7 | | | |

Note: To write data into the I2C data shift register after setting the MST bit to "0" (slave mode), keep an interval of 8 machine cycles or more.

**Figure 11.3 I2C data shift register**

I2C transmit buffer register

| Symbol | Address | When reset |
|---|---|---|
| IIC0S0S | 02E616 | Indeterminate |

b7 b6 b5 b4 b3 b2 b1 b0

| Bit Symbol | Bit name | Function | RW |
|---|---|---|---|
| S0S0 | Transmit buffer register | This is an 8-bit register to write transmit data to I2C data shift register. | WO |
| S0S1 | | | |
| S0S2 | | | |
| S0S3 | | | |
| S0S4 | | | |
| S0S5 | | | |
| S0S6 | | | |
| S0S7 | | | |

**Figure 11.4 I2C transmit buffer register**

### (3) I2C address register

The I2C address register consists of a 7-bit slave address and a read/write bit. In the addressing mode, the slave address written in this register is compared with the address data to be received immediately after the START condition are detected.

**• Bit 0: read/write bit (RBW)**

Not used when comparing addresses, in the 7-bit addressing mode. In the 10-bit addressing mode, the first address data to be received is compared with the contents (SAD6 to SAD0 + RBW) of the I2C address register.
The RBW bit is cleared to "0" automatically when the stop condition is detected.

**• Bits 1 to 7: slave address (SAD0 to SAD6)**

These bits store slave addresses. Regardless of the 7-bit addressing mode and the 10-bit addressing mode, the address data transmitted from the master is compared with the contents of these bits.

I2C address register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Symbol: IIC0S0D
Address: 02E116
When reset: 0016

| Bit Symbol | Bit name | Function | RW |
|---|---|---|---|
| RBW | Read/write bit | <Only in 10-bit addressing (in slave) mode><br>The last significant bit of address data is compared.<br>0 : Wait the first byte of slave address after START condition (read state)<br>1 : Wait the first byte of slave address after RESTART condition (write state) | RW |
| SAD0 | Slave address | <In both modes><br>The address data is compared. | RW |
| SAD1 | | | |
| SAD2 | | | |
| SAD3 | | | |
| SAD4 | | | |
| SAD5 | | | |
| SAD6 | | | |

**Figure 11.5 I2C address register**

### (4) I2C clock control register

The I2C clock control register is used to set ACK control, SCL mode and SCL frequency.

**• Bits 0 to 4: SCL frequency control bits (CCR0–CCR4)**

These bits control the SCL frequency.

**• Bit 5: SCL mode specification bit (FAST MODE)**

This bit specifies the SCL mode. When this bit is set to "0," the standard clock mode is set. When the bit is set to "1," the high-speed clock mode is set.

**• Bit 6: ACK bit (ACK BIT)**

This bit sets the SDA status when an ACK clock∗ is generated. When this bit is set to "0," the ACK return mode is set and SDA goes to LOW at the occurrence of an ACK clock. When the bit is set to "1," the ACK non-return mode is set. The SDA is held in the HIGH status at the occurrence of an ACK clock.

However, when the slave address matches the address data in the reception of address data at ACK BIT = "0," the SDA is automatically made LOW (ACK is returned). If there is a mismatch between the slave address and the address data, the SDA is automatically made HIGH (ACK is not returned).

∗ACK clock: Clock for acknowledgement

**• Bit 7: ACK clock bit (ACK)**

This bit specifies a mode of acknowledgment which is an acknowledgment response of data transmission.

When this bit is set to "0," the no ACK clock mode is set. In this case, no ACK clock occurs after data transmission. When the bit is set to "1," the ACK clock mode is set and the master generates an ACK clock upon completion of each 1-byte data transmission.The device for transmitting address data and control data releases the SDA at the occurrence of an ACK clock (make SDA HIGH) and receives the ACK bit generated by the data receiving device.

**Note:** Do not write data into the I2C clock control register during transmission. If data is written during transmission, the I2C clock generator is reset, so that data cannot be transmitted normally.

I2C clock control register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Symbol: IIC0S2 — Address: 02E4₁₆ — When reset: 00₁₆

| Bit Symbol | Bit name | Function | | | RW |
|---|---|---|---|---|---|
| CCR0 | SCL frequency control bits | Setup value of CCR4–CCR0 | Standard clock mode | High speed clock mode | RW |
| CCR1 | | 00 to 02 | Setup disabled | Setup disabled | |
| CCR2 | | 03 | Setup disabled | 333 | |
| CCR3 | | 04 | Setup disabled | 250 | |
| CCR4 | | 05 | 100 | 400 (See note) | |
| | | 06 | 83.3 | 166 | |
| | | : | 500/CCR value | 1000/CCR value | |
| | | 1D | 17.2 | 34.5 | |
| | | 1E | 16.6 | 33.3 | |
| | | 1F | 16.1 | 32.3 | |
| | | (at BCLK = 10 MHz, unit : kHz) | | | |
| FAST MODE | SCL mode specification bit | 0 : Standard clock mode<br>1 : High-speed clock mode | | | RW |
| ACK BIT | ACK bit | 0 : ACK is returned.<br>1 : ACK is not returned. | | | RW |
| ACK | ACK clock bit | 0 : No ACK clock<br>1 : ACK clock | | | RW |

**Note:** At 400 kHz in the high-speed clock mode, the duty is as below.
"0" period : "1" period = 3 : 2
In the other cases, the duty is as below.
"0" period : "1" period = 1 : 1

**Figure 11.6 I2C clock control register**

### (5) I2C control register

The I2C control register controls the data communication format.

**• Bits 0 to 2: bit counter (BC0–BC2)**

These bits decide the number of bits for the next 1-byte data to be transmitted. An interrupt request signal occurs immediately after the number of bits specified with these bits are transmitted.

When a START condition is received, these bits become "0002" and the address data is always transmitted and received in 8 bits.

**Note:** When the bit counter value = "1112," a STOP condition and START condition cannot be waited.

**• Bit 3: I2C-BUS interface use enable bit (ESO)**

This bit enables usage of the multimaster I2C-BUS interface i. When this bit is set to "0," the use disable status is provided, so the SDA and the SCL become high-impedance. When the bit is set to "1," use of the interface is enabled.

When ESO = "0," the following is performed.

- PIN = "1," BB = "0" and AL = "0" are set (they are bits of the I2C status register).
- Writing data to the I2C data shift register and the I2C transmit buffer register is disabled.

**• Bit 4: data format selection bit (ALS)**

This bit decides whether or not to recognize slave addresses. When this bit is set to "0," the addressing format is selected, so that address data is recognized. When a match is found between a slave address and address data as a result of comparison or when a general call (refer to "(6) I2C status register," bit 1) is received, transmission processing can be performed. When this bit is set to "1," the free data format is selected, so that slave addresses are not recognized.

**• Bit 5: addressing format selection bit (10BIT SAD)**

This bit selects a slave address specification format. When this bit is set to "0," the 7-bit addressing format is selected. In this case, only the high-order 7 bits (slave address) of the I2C address register are compared with address data. When this bit is set to "1," the 10-bit addressing format is selected, all the bits of the I2C address register are compared with address data.

I2C control register

| Symbol | Address | When reset |
|---|---|---|
| IIC0S1D | 02E3₁₆ | 00₁₆ |

b7 b6 b5 b4 b3 b2 b1 b0

| Bit Symbol | Bit name | Function | RW |
|---|---|---|---|
| BC0 | Bit counter (Number of transmit/receive bits) | b2 b1 b0<br>0 0 0 : 8<br>0 0 1 : 7<br>0 1 0 : 6<br>0 1 1 : 5<br>1 0 0 : 4<br>1 0 1 : 3<br>1 1 0 : 2<br>1 1 1 : 1 | RW |
| BC1 | | | |
| BC2 | | | |
| ESO | I2C-BUS interface use enable bit | 0 : Disabled<br>1 : Enabled | RW |
| ALS | Data format selection bit | 0 : Addressing format<br>1 : Free data format | RW |
| 10BIT SAD | Address format selection bit | 0 : 7-bit addressing format<br>1 : 10-bit addressing format | RW |
| Nothing is assigned.<br>In an attempt to write to these bits, write “0.”<br>The value, if read, turns out to be “0.” | | | — |

**Figure 11.7 I2C control register**

### (6) I2C status register

The I2C status register controls the I2C-BUS interface status. Bits 0 to 3, 5 are read-only bits and bits 4, 6, 7 can be read out and written to.

**• Bit 0: last receive bit (LRB)**

This bit stores the last bit value of received data and can also be used for ACK receive confirmation. If ACK is returned when an ACK clock occurs, the LRB bit is set to "0." If ACK is not returned, this bit is set to "1." Except in the ACK mode, the last bit value of received data is input. The state of this bit is changed from "1" to "0" by executing a write instruction to the I2C data shift register or the I2C transmit buffer register.

**• Bit 1: general call detecting flag (AD0)**

This bit is set to "1" when a general call* whose address data is all "0" is received in the slave mode.
By a general call of the master device, every slave device receives control data after the general call.
The AD0 bit is set to ì0î by detecting the STOP condition or START condition.

*General call: The master transmits the general call address "0016"to all slaves.

**• Bit 2: slave address comparison flag (AAS)**

This flag indicates a comparison result of address data.

<<In the slave receive mode, when the 7-bit addressing format is selected, this bit is set to "1" in one of the following conditions.>>

- The address data immediately after occurrence of a START condition matches the slave address stored in the high-order 7 bits of the I2C address register.
- A general call is received.

<<In the slave reception mode, when the 10-bit addressing format is selected, this bit is set to "1" with the following condition.>>

- When the address data is compared with the I2C address register (8 bits consists of slave address and RBW), the first bytes match.

<<The state of this bit is changed from "1" to "0" by executing a write instruction to the I2C data shift register or the I2C transmit buffer register.>>

**• Bit 3: arbitration lost* detecting flag (AL)**

In the master transmission mode, when a device other than the microcomputer sets the SDA to "L," arbitration is judged to have been lost, so that this bit is set to "1." At the same time, the TRX bit is set to "0," so that immediately after transmission of the byte whose arbitration was lost is completed, the MST bit is set to "0." When arbitration is lost during slave address transmission, the TRX bit is set to "0" and the reception mode is set. Consequently, it becomes possible to receive and recognize its own slave address transmitted by another master device.

<<This bit changes "1" to "0" by writing instruction to I2C data shift register or I2C transmit buffer register.>>

*Arbitration lost: The status in which communication as a master is disabled.

**• Bit 4: I2C-BUS interface interrupt request bit (PIN)**

This bit generates an interrupt request signal. Each time 1-byte data is transmitted, the state of the PIN bit changes from "1" to "0." At the same time, an interrupt request signal is sent to the CPU. The PIN bit is set to "0" in synchronization with a falling edge of the last clock (including the ACK clock) of an internal clock and an interrupt request signal occurs in synchronization with a falling edge of the PIN bit. When detecting the STOP condition in slave, the multi-master I2C-BUS interface interrupt request bit (IR) is set to "1" (interrupt requested) regardless of falling of PIN bit. When the PIN bit is "0," the SCL is kept in the "0" state and clock generation is disabled. Figure 11.9 shows an interrupt request signal generating timing chart.

The PIN bit is set to "1" in any one of the following conditions.

- Writing "1" to the PIN bit
- Executing a write instruction to the I2C data shift register or the I2C transmit buffer register (See note).
- When the ESO bit is "0"
- At reset

**Note:** It takes 12 BCLK cycles or more until PIN bit becomes "1" after write instructions are executed to these registers.

The conditions in which the PIN bit is set to "0" are shown below:

- Immediately after completion of 1-byte data transmission (including when arbitration lost is detected)
- Immediately after completion of 1-byte data reception
- In the slave reception mode, with ALS = "0" and immediately after completion of slave address or general call address reception
- In the slave reception mode, with ALS = "1" and immediately after completion of address data reception

**• Bit 5: bus busy flag (BB)**

This bit indicates the status of use of the bus system. When this bit is set to "0," this bus system is not busy and a START condition can be generated. When this bit is set to "1," this bus system is busy and the occurrence of a START condition is disabled by the START condition duplication prevention function (See note).

This flag cannot be written with software. In the other modes, this bit is set to "1" by detecting a START condition and set to "0" by detecting a STOP condition. When the ESO bit of the I2C control register is "0" and at reset, the BB flag is kept in the "0" state.

**• Bit 6: communication mode specification bit (transfer direction specification bit: TRX)**

This bit decides the direction of transfer for data communication. When this bit is "0," the reception mode is selected and the data of a transmitting device is received. When the bit is "1," the transmission mode is selected and address data and control data are output into the SDA in synchronization with the clock generated on the SCL.

When the ALS bit of the I2C control register is "0" in the slave reception mode is selected, the TRX bit is set to "1" (transmit) if the least significant bit (R/W bit) of the address data transmitted by the master is "1." When the ALS bit is "0" and the R/W bit is "0," the TRX bit is cleared to "0" (receive).

The TRX bit is cleared to "0" in one of the following conditions.

- When arbitration lost is detected.
- When a STOP condition is detected.
- When occurrence of a START condition is disabled by the START condition duplication prevention function (Note).
- With MST = "0" and when a START condition is detected.
- With MST = "0" and when ACK non-return is detected.
- At reset

**• Bit 7: Communication mode specification bit (master/slave specification bit: MST)**

This bit is used for master/slave specification for data communication. When this bit is "0," the slave is specified, so that a START condition and a STOP condition generated by the master are received, and data communication is performed in synchronization with the clock generated by the master.

When this bit is "1," the master is specified and a START condition and a STOP condition are generated, and also the clocks required for data communication are generated on the SCL.

The MST bit is cleared to "0" in one of the following conditions.

- Immediately after completion of 1-byte data transmission when arbitration lost is detected
- When a STOP condition is detected.
- When occurence of a START condition is disabled by the START condition duplication preventing function (See note).
- At reset

**Note:** The START condition duplication prevention function disables the following: the START condition generation; bit counter reset, and SCL output with the generation. This bit is valid from setting of BB flag to the completion of 1-byte transmission/reception (occurrence of transmission/ reception interrupt request) <IICRQ>.

I2C status register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Symbol: IIC0S1 — Address: 02E2$_{16}$ — When reset: 0001000?$_2$

| Bit Symbol | Bit name | Function | RW |
|---|---|---|---|
| LRB | Last receive bit | 0 : Last bit = "0"<br>1 : Last bit = "1" (See note 1) | RO |
| AD0 | General call detecting flag | 0 : No general call detected<br>1 : General call detected (See note 1) | RO |
| AAS | Slave address comparison flag | 0 : Address mismatch<br>1 : Address match (See note 1) | RO |
| AL | Arbitration lost detecting flag | 0 : Not detected<br>1 : Detected (See note 1) | RO |
| PIN | I2C-BUS interface interrupt request bit | 0 : Interrupt request issued<br>1 : No interrupt request issued (See note 2) | RW |
| BB | Bus busy flag | 0 : Bus free<br>1 : Bus busy (See note 1) | RO |
| TRX | Communication mode specification bits | b7b6<br>0 0 : Slave receive mode<br>0 1 : Slave transmit mode<br>1 0 : Master receive mode<br>1 1 : Master transmit mode | RW |
| MST | | | |

**Notes 1:** These bits and flags can be read out, but cannot be written.
**2:** This bit can be written only "1."

**Figure 11.8 I2C status register**

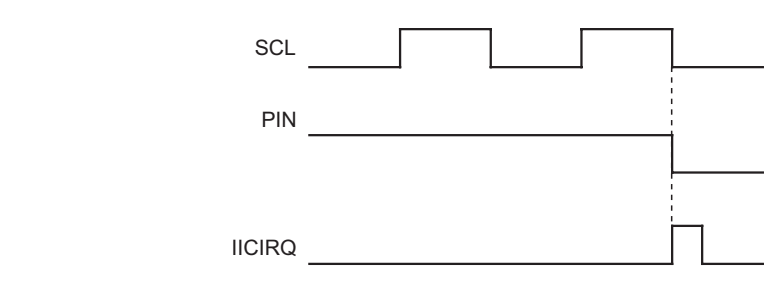



**Figure 11.9 Interrupt request signal generation timing**

### (7) START condition generation method

When the ESO bit of the I2C control register is "1," execute a write instruction to the I2C status register to set the MST, TRX and BB bits to "1." A START condition will then be generated. After that, the bit counter becomes "0002" and an SCL for 1 byte is output. The START condition generation timing and BB bit set timing are different in the standard clock mode and the high-speed clock mode. Refer to Figure 11.10 for the START condition generation timing diagram, and Table 11.2 for the START condition/STOP condition generation timing table.

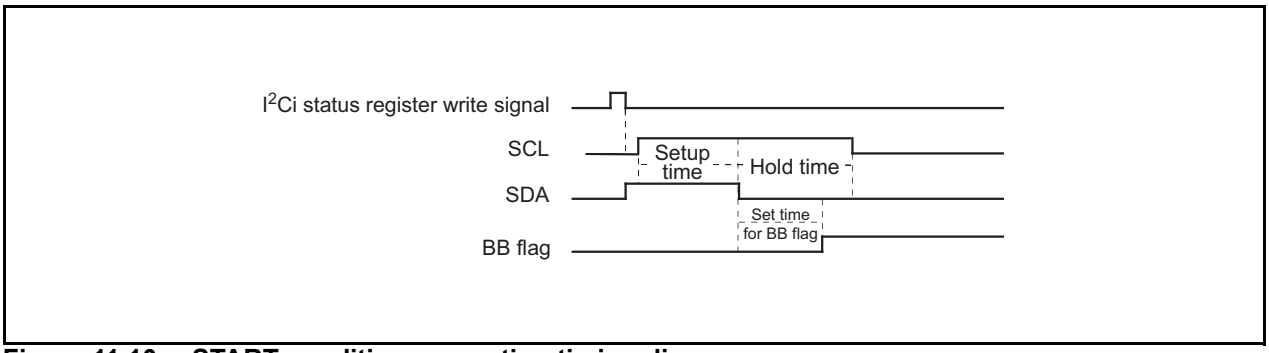



**Figure 11.10 START condition generation timing diagram**

### (8) STOP condition generation method

When the ESO bit of the I2C control register is "1," execute a write instruction to the I2C status register for setting the MST bit and the TRX bit to "1" and the BB bit to "0". A STOP condition will then be generated. The STOP condition generation timing and the BB flag reset timing are different in the standard clock mode and the high-speed clock mode. Refer to Figure 11.11 for the STOP condition generation timing diagram, and Table 11.2 for the START condition/STOP condition generation timing table.

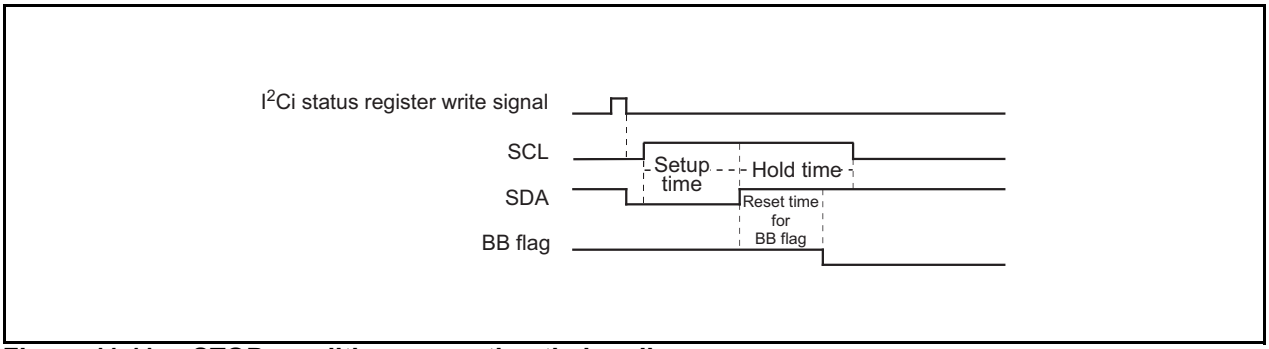



**Figure 11.11 STOP condition generation timing diagram**

**Table 11.2 START condition/STOP condition generation timing table**

| Item | Standard Clock Mode | High-speed Clock Mode |
|---|---|---|
| Setup time (Min.) | 5.6 µs | 2.1 µs |
| Hold time (Min.) | 4.8 µs | 2.3 µs |
| Set/reset time for BB flag | 3.5 µs | 0.75 µs |

### (9) START/STOP condition detect conditions

The START/STOP condition detect conditions are shown in Figure 11.12 and Table 11.3.
Only when the 3 conditions of Table 11.3 are satisfied, a START/STOP condition can be detected.

**Note:** When a STOP condition is detected in the slave mode (MST = 0), an interrupt request signal <IICRQ> is generated to the CPU.

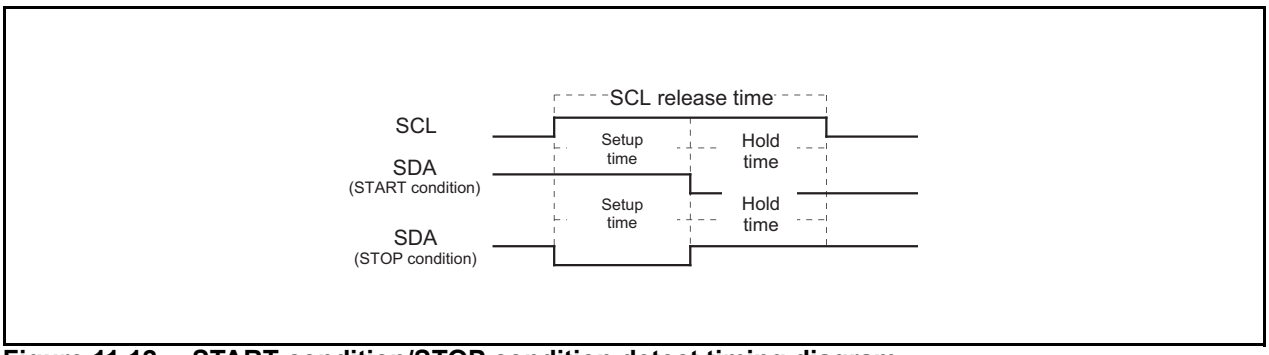



**Figure 11.12 START condition/STOP condition detect timing diagram**

**Table 11.3 START condition/STOP condition detect conditions**

| Standard Clock Mode | High-speed Clock Mode |
|---|---|
| 6.5 μs < SCL release time | 1.0 μs < SCL release time |
| 3.25 μs < Setup time | 0.5 μs < Setup time |
| 3.25 μs < Hold time | 0.5 μs < Hold time |

### (10) Address data communication

There are two address data communication formats, namely, 7-bit addressing format and 10-bit addressing format. The respective address communication formats is described below.

**• 7-bit addressing format**

To meet the 7-bit addressing format, set the 10BIT SAD bit of the I2C control register to "0." The first 7-bit address data transmitted from the master is compared with the high-order 7-bit slave address stored in the I2C address register. At the time of this comparison, address comparison of the RBW bit of the I2C address register is not made. For the data transmission format when the 7-bit addressing format is selected, refer to Figure 11.13 (1) and (2).

**• 10-bit addressing format**

To meet the 10-bit addressing format, set the 10BIT SAD bit of the I2C control register to "1." An address comparison is made between the first-byte address data transmitted from the master and the 7-bit slave address stored in the I2C address register. At the time of this comparison, an address comparison between the RBW bit of the I2C address register and the R/$\overline{W}$ bit which is the last bit of the address data transmitted from the master is made. In the 10-bit addressing mode, the R/$\overline{W}$ bit which is the last bit of the address data not only specifies the direction of communication for control data but also is processed as an address data bit.

When the first-byte address data matches the slave address, the AAS bit of the I2C status register is set to "1." After the second-byte address data is stored into the I2C data shift register, make an address comparison between the second-byte data and the slave address by software. When the address data of the 2nd bytes matches the slave address, set the RBW bit of the I2C address register to "1" by software. This processing can match the 7-bit slave address and R/$\overline{W}$ data, which are received after a RESTART condition is detected, with the value of the I2C address register. For the data transmission format when the 10-bit addressing format is selected, refer to Figure 11.13, (3) and (4).

### (11) Example of Master Transmission

An example of master transmission in the standard clock mode, at the SCL frequency of 100 kHz and in the ACK return mode is shown below.

(1) Set a slave address in the high-order 7 bits of the I2C address register and "0" in the RBW bit.
(2) Set the ACK return mode and SCL = 100 kHz by setting "8516" in the I2C clock control register.
(3) Set "1016" in the I2C status register and hold the SCL at the HIGH.
(4) Set a communication enable status by setting "0816" in the I2C control register.
(5) Set the address data of the destination of transmission in the high-order 7 bits of the I2C data shift register and set "0" in the least significant bit.
(6) Set "F016" in the I2C status register to generate a START condition. At this time, an SCL for 1 byte and an ACK clock automatically occurs.
(7) Set transmit data in the I2C data shift register. At this time, an SCL and an ACK clock automatically occurs.
(8) When transmitting control data of more than 1 byte, repeat step (7).
(9) Set "D016" in the I2C status register. After this, if ACK is not returned or transmission ends, a STOP condition will be generated.

### (12) Example of Slave Reception

An example of slave reception in the high-speed clock mode, at the SCL frequency of 400 kHz, in the ACK non-return mode, using the addressing format, is shown below.

(1) Set a slave address in the high-order 7 bits of the I2C address register and "0" in the RBW bit.
(2) Set the no ACK clock mode and SCL = 400 kHz by setting "2516" in the I2C clock control register.
(3) Set "1016" in the I2C status register and hold the SCL at the HIGH.
(4) Set a communication enable status by setting "0816" in the I2C control register.
(5) When a START condition is received, an address comparison is made.
(6)
•When all transmitted address are"0" (general call):
AD0 of the I2C status register is set to "1"and an interrupt request signal occurs.
•When the transmitted addresses match the address set in (1):
ASS of the I2C status register is set to "1" and an interrupt request signal occurs.
•In the cases other than the above:
AD0 and AAS of the I2C status register are set to "0" and no interrupt request signal occurs.
(7) Set dummy data in the I2C data shift register.
(8) When receiving control data of more than 1 byte, repeat step (7).
(9) When a STOP condition is detected, the communication ends.

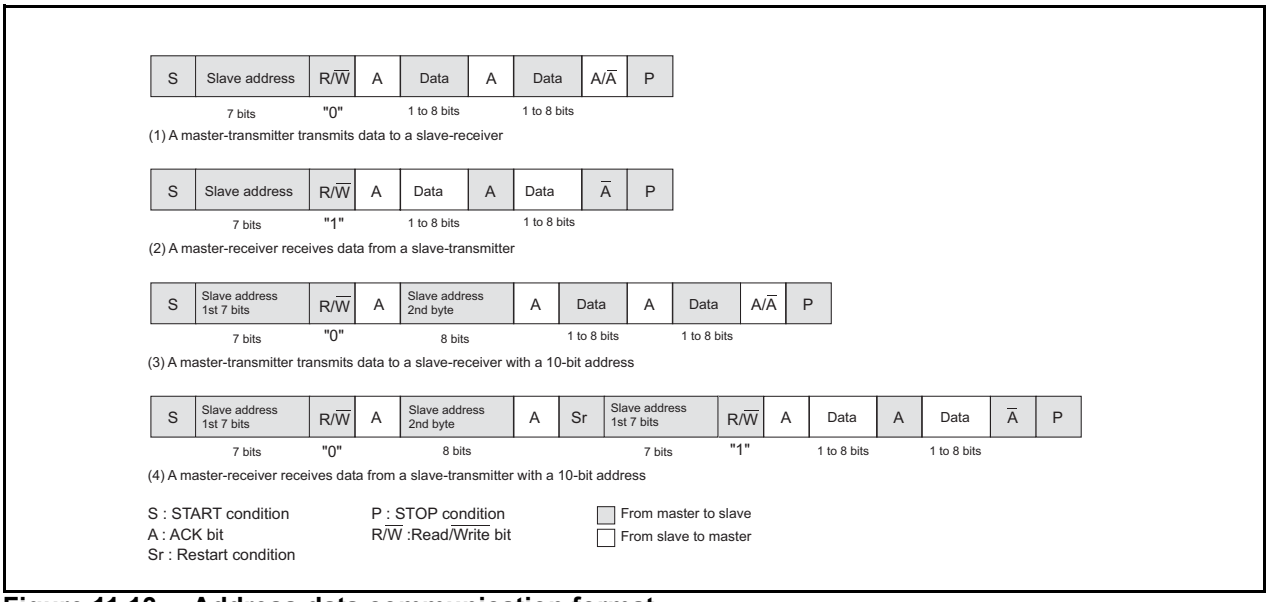



**Figure 11.13 Address data communication format**

### (13) Precautions when using multi-master I2C-BUS interface

- **BCLK operation mode**
  Select the no-division mode.
- **Used instructions**
  Specify byte (.B) as data size to access multi-master I2C-BUS interface i-related registers.
- **Read-modify-write instruction**
  The precautions when the read-modify-write instruction such as BSET, BCLR etc. is executed for each register of the multi-master I2C-BUS interface are described below.
  - I2C data shift register (IICS0)
    When executing the read-modify-write instruction for this register during transfer, data may become a value not intended.
  - I2C address register (IICS0D)
    When the read-modify-write instruction is executed for this register at detecting the STOP condition, data may become a value not intended. It is because hardware changes the read/write bit (RBW) at the above timing.
  - I2C status register (IICS1)
    Do not execute the read-modify-write instruction for this register because all bits of this register are changed by hardware.
  - I2C control register (IICS1D)
    When the read-modify-write instruction is executed for this register at detecting the START condition or at completing the byte transfer, data may become a value not intended. Because hardware changes the bit counter (BC0–BC2) at the above timing.
  - I2C clock control register (IICS2)
    The read-modify-write instruction can be executed for this register.
  - I2C port selection register (IICS2D)
    Since the read value of high-order 4 bits is indeterminate, the read-modify-write instruction cannot be used.
  - I2C transmit buffer register (IICS0S)
    Since the value of all bits is indeterminate, the read-modify-write instruction cannot be used.

**• START condition generating procedure using multi-master**

```
                :
        FCLR    I               (Interrupt disabled)
        BTST    5, IICS1        (BB flag confirming and branch process)
        JC      BUSBUSY
BUSFREE:
        MOV.B   SA, IICS0       (Writing of slave address value <SA>)
        NOP
        NOP
        NOP
        NOP
        MOV.B   #F0H, IICS1     (Trigger of START condition generating)
        FSET    I               (Interrupt enabled)
                :
BUSBUSY:
        FSET    I               (Interrupt enabled)
                :
```

(1) and (2)

(1) Be sure to add NOP instruction × 4 between writing the slave address value and setting trigger of START condition generating shown the above procedure example.

(2) When using multi-master system, disable interrupts during the following three process steps:
- BB flag confirming
- Writing of slave address value
- Trigger of START condition generating

When the condition of the BB flag is bus busy, enable interrupts immediately.
When using single-master system, it is not necessary to disable interrupts above.

**• RESTART condition generating procedure**

```
        :
MOV.B   SA, IICS0S      (Writing of slave address value <SA>)
NOP                                                            (1)
NOP
MOV.B   #F0H, IICS1     (Trigger of RESTART condition generating)
        :
```

(1) Use the I2C transmit buffer register to write the slave address value to the I2C data shift register. And also, be sure to add NOP instruction × 4.

**• Writing to I2C status register**

Do not execute an instruction to set the PIN bit to "1" from "0" and an instruction to set the MST and TRX bits to "0" from "1" simultaneously. It is because it may enter the state that the SCL pin is released and the SDA pin is released after about one machine cycle. Do not execute an instruction to set the MST and TRX bits to "0" from "1" simultaneously when the PIN bit is "1." It is because it may become the same as above.

**• Process of after STOP condition generating**

Do not write data in the I2C data shift register (IICS0) and the I2C status register (IICS1) until the bus busy flag BB becomes "0" after generating the STOP condition in the master mode. It is because the STOP condition waveform might not be normally generated. Reading to the above registers do not have the problem.

## I2C0 Interrupt Control Register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | 0 | 0 | 0 | 0 |

Symbol: EXTIICINT — Address: 02D616 — At reset: 0016

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Reserved bits | Must set to "0." | | ○ | ○ |
| EXTIICINT0 | ACK interrupt control bit (Note 1) | 0000: Interrupt prohibition (Note 2)<br>0101: Interrupt permission<br>Other: Must not be set<br><br>Please set TA4IC (Note 3) when you use it by "Interrupt permission". | ○ | ○ |
| EXTIICINT1 | | | | |
| EXTIICINT2 | | | | |
| EXTIICINT3 | | | | |

Notes 1: Timer A4 and multi master I2C (ACK) interrupt, the vector and the interrupt control register are shared. Please make it to (b7, b6, b5, b4) = (0, 1, 0, 1) when you use multi-master I2C (ACK) interrupt.
Notes 2: Please set $0000_2$ when you use the interrupt of timer A4.
Notes 3: Please refer to "Figure 6.3 interrupt control register" of "6.5 interrupt control".
Notes 4: Please change in the part where multi-master I2C (ACK) and timer A4 interrupt request are not generated in the I2C0 interrupt control register.
Notes 5: Please permit interrupt after making IR bit of timer A4 (TA4IC) "0" (the interrupt request none) after EXTIICINTi (i = 0 to 3) is changed.

## Reserved Register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | | 0 | | 0 |

Symbol: RSVREG02D7 — Address: 02D716 — At reset: 0016

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Reserved bit | Must set to "0." | | ○ | ○ |
| Reserved bit | When use multi-master I2C-BUS interface, set this bit to "1." | | ○ | ○ |
| Reserved bit | Must set to "0." | | ○ | ○ |
| Reserved bit | When use multi-master I2C-BUS interface, set this bit to "1." | | ○ | ○ |
| Reserved bits | Must set to "0." | | ○ | ○ |

# 12. A/D Converter

The microcomputer contains one A/D converter circuit based on 8-bit successive approximation method configured with a capacitive-coupling amplifier. The analog inputs share the pins with P0$_0$ to P0$_7$, P9$_5$ and P9$_6$. Similarly, $\overline{\text{ADTRG}}$ input shares the pin with P9$_7$. Therefore, when using these inputs, make sure the corresponding port direction bits are set to "0" (= input mode).
When not using the A/D converter, set the VCUT bit to "0" (= Vref unconnected), so that no current will flow from the $V_{REF}$ pin into the resistor ladder, helping to reduce the power consumption of the chip.
The A/D conversion result is stored in the ADi register bits for ANi pins (i = 0 to 7).
Table 12.1 shows the performance of the A/D converter. Figure 12.1 shows the block diagram of the A/D converter, and Figures 12.2 and 12.3 show the A/D converter-related registers.

**Table 12.1 Performance of A/D Converter**

| Item | Performance |
|---|---|
| Method of A/D conversion | Successive approximation (capacitive coupling amplifier) |
| Analog input voltage (Note 1) | 0V to $AV_{CC}$ ($V_{CC}$) |
| Operating clock $\phi_{AD}$ (Note 2) | $f_{AD}$/divide-by-2 of $f_{AD}$/divide-by-3 of $f_{AD}$/divide-by-4 of $f_{AD}$/divide-by-6 of $f_{AD}$/divide-by-12 of $f_{AD}$ |
| Resolution | 8-bit |
| Integral nonlinearity error | When $AV_{CC}$ = $V_{REF}$ = 5V<br>• With 8-bit resolution: ±3LSB<br>- ANEX0 and ANEX1 input (including mode in which external operation amp is connected) : ±4LSB |
| Operating modes | One-shot mode, repeat mode, single sweep mode, repeat sweep mode 0, and repeat sweep mode 1 |
| Analog input pins | 8 pins (AN$_0$ to AN$_7$) + 2 pins (ANEX0 and ANEX1) |
| A/D conversion start condition | • Software trigger<br>The ADCON0 register's ADST bit is set to "1" (A/D conversion starts)<br>• External trigger (retriggerable)<br>Input on the $\overline{\text{ADTRG}}$ pin changes state from high to low after the ADST bit is set to "1" (A/D conversion starts) |
| Conversion speed per pin | • Without sample and hold function<br>8-bit resolution: 49 $\phi_{AD}$ cycles<br>• With sample and hold function<br>8-bit resolution: 28 $\phi_{AD}$ cycles |

Note 1: Does not depend on use of sample and hold function.
Note 2: The $\phi_{AD}$ frequency must be 10 MHz or less.
Without sample-and-hold function, limit the $\phi_{AD}$ frequency to 250kHz or more.
With the sample and hold function, limit the $\phi_{AD}$ frequency to 1MHz or more.

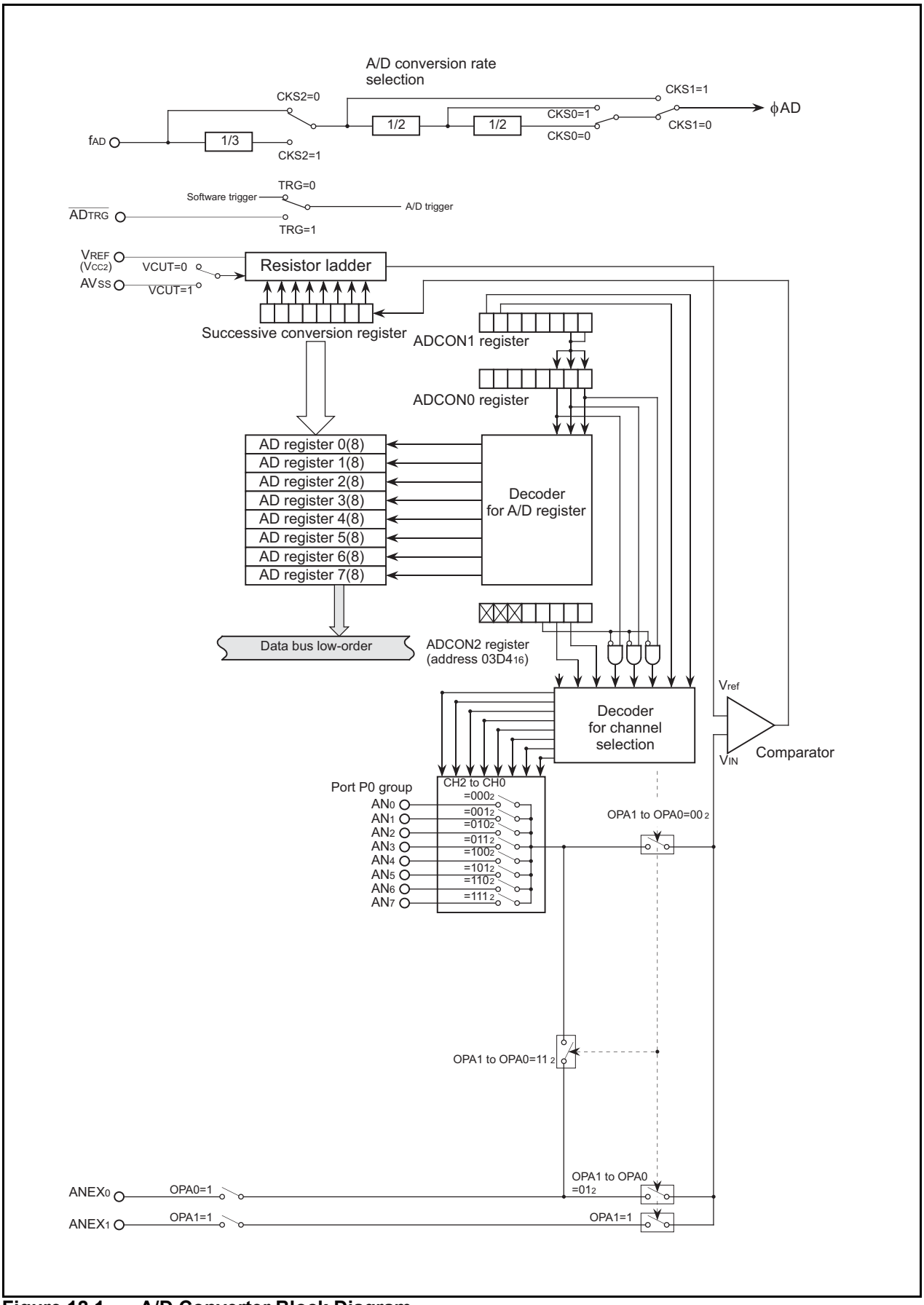
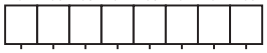



**Figure 12.1 A/D Converter Block Diagram**

A/D control register 0 (Note)

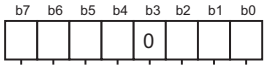



Symbol ADCON0 | Address 03D6₁₆ | After reset 00000XXX₂

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| CH0 | Analog input pin select bit | Function varies with each operation mode | RW |
| CH1 | | | RW |
| CH2 | | | RW |
| MD0 | A/D operation mode select bit 0 | b4 b3<br>0 0 : One-shot mode<br>0 1 : Repeat mode<br>1 0 : Single sweep mode<br>1 1 : Repeat sweep mode 0 or Repeat sweep mode 1 | RW |
| MD1 | | | RW |
| TRG | Trigger select bit | 0 : Software trigger<br>1 : ADTRG trigger | RW |
| ADST | A/D conversion start flag | 0 : A/D conversion disabled<br>1 : A/D conversion started | RW |
| CKS0 | Frequency select bit 0 | See Note 2 for the ADCON2 register | RW |

Note: If the ADCON0 register is rewritten during A/D conversion, the conversion result will be indeterminate.

A/D control register 1 (Note 1)



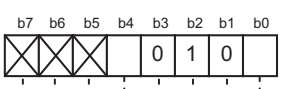

Symbol ADCON1 | Address 03D7₁₆ | After reset 00₁₆

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| SCAN0 | A/D sweep pin select bit | Function varies with each operation mode | RW |
| SCAN1 | | | RW |
| MD2 | A/D operation mode select bit 1 | 0 : Any mode other than repeat sweep mode 1<br>1 : Repeat sweep mode 1 | RW |
| $\overline{(b3)}$ | Reserved bit | Must always be set to "0" | RW |
| CKS1 | Frequency select bit 1 | See Note 2 for the ADCON2 register | RW |
| VCUT | Vref connect bit (Note 2) | 0 : Vref not connected<br>1 : Vref connected | RW |
| OPA0 | External op-amp connection mode bit | Function varies with each operation mode | RW |
| OPA1 | | | RW |

Note 1: If the ADCON1 register is rewritten during A/D conversion, the conversion result will be indeterminate.
Note 2: If the VCUT bit is reset from "0" (Vref unconnected) to "1" (Vref connected), wait for 1 µs or more before starting A/D conversion.

**Figure 12.2 ADCON0 to ADCON1 Registers**

A/D control register 2 (Note 1)




| Symbol | Address | After reset |
|---|---|---|
| ADCON2 | 03D4₁₆ | 00₁₆ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| SMP | A/D conversion method select bit | 0 : Without sample and hold<br>1 : With sample and hold | RW |
| (b1) | Reserved bit | Must always be set to "0" | RW |
| (b2) | Reserved bit | Must always be set to "1" | RW |
| (b3) | Reserved bit | Must always be set to "0" | RW |
| CKS2 | Frequency select bit 2 (Note 2) | 0: Selects fAD, fAD divided by 2, or fAD divided by 4.<br>1: Selects fAD divided by 3, fAD divided by 6, or fAD divided by 12. | RW |
| (b7-b5) | Nothing is assigned. In an attempt to write to these bits, write "0". The value, if read, turns out to be "0". | | — |

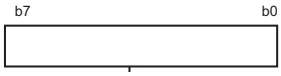
Note 1: If the ADCON2 register is rewritten during A-D conversion, the conversion result will be indeterminate.

Note 2: Adjust the frequency of ϕAD to 10MHZ or less. ϕAD can be selected by combining CKS0 bit of ADCON0 register, CKS1 bit of ADCON1 register and CKS2 bit of ADCON2 register.

| CKS2 | CKS1 | CKS0 | ϕAD |
|---|---|---|---|
| 0 | 0 | 0 | Divide-by-4 of fAD |
| 0 | 0 | 1 | Divide-by-2 of fAD |
| 0 | 1 | 0 | fAD |
| 0 | 1 | 1 | |
| 1 | 0 | 0 | Ddivide-by-12 of fAD |
| 1 | 0 | 1 | Divide-by-6 of fAD |
| 1 | 1 | 0 | Divide-by-3 of fAD |
| 1 | 1 | 1 | |

A/D register i (i=0 to 7)

| Symbol | Address | After reset |
|---|---|---|
| AD0 | 03C0₁₆ | Indeterminate |
| AD1 | 03C2₁₆ | Indeterminate |
| AD2 | 03C4₁₆ | Indeterminate |
| AD3 | 03C6₁₆ | Indeterminate |
| AD4 | 03C8₁₆ | Indeterminate |
| AD5 | 03CA₁₆ | Indeterminate |
| AD6 | 03CC₁₆ | Indeterminate |
| AD7 | 03CE₁₆ | Indeterminate |




| Function | RW |
|---|---|
| A/D conversion result | RO |

**Figure 12.3 ADCON2 Register, and AD0 to AD7 Registers**

## 12.1 One-shot Mode

In this mode, the input voltage on one selected pin is A/D converted once. Table 12.2 shows the specifications of one-shot mode. Figure 12.4 shows the ADCON0 to ADCON1 registers in one-shot mode.

**Table 12.2 One-shot Mode Specifications**

| Item | Specification |
|---|---|
| Function | The input voltage on one pin selected by the ADCON0 register's CH2 to CH0 bits and the ADCON1 register's OPA1 to OPA0 bits is A/D converted once. |
| A/D conversion start condition | • When the ADCON0 register's TRG bit is “0” (software trigger)<br>The ADCON0 register's ADST bit is set to “1” (A/D conversion starts)<br>• When the TRG bit is “1” ($\overline{\text{ADTRG}}$ trigger)<br>Input on the $\overline{\text{ADTRG}}$ pin changes state from high to low after the ADST bit is set to “1” (A/D conversion starts) |
| A/D conversion stop condition | • Completion of A/D conversion (If a software trigger is selected, the ADST bit is cleared to “0” (A/D conversion halted).)<br>• Set the ADST bit to “0” |
| Interrupt request generation timing | Completion of A/D conversion |
| Analog input pin | Select one pin from $AN_0$ to $AN_7$, ANEX0 to ANEX1 |
| Reading of result of A/D converter | Read one of the AD0 to AD7 registers that corresponds to the selected pin |

A/D control register 0 (Note 1)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | 0 | 0 | | | |

| Symbol | Address | After reset |
|---|---|---|
| ADCON0 | $03D6_{16}$ | $00000XXX_2$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| CH0 | Analog input pin select bit | b2 b1 b0<br>0 0 0 : $AN_0$ is selected<br>0 0 1 : $AN_1$ is selected<br>0 1 0 : $AN_2$ is selected<br>0 1 1 : $AN_3$ is selected<br>1 0 0 : $AN_4$ is selected<br>1 0 1 : $AN_5$ is selected<br>1 1 0 : $AN_6$ is selected<br>1 1 1 : $AN_7$ is selected (Note 2) | RW |
| CH1 | | | RW |
| CH2 | | | RW |
| MD0 | A/D operation mode select bit 0 | b4 b3<br>0 0 : One-shot mode (Note 2) | RW |
| MD1 | | | RW |
| TRG | Trigger select bit | 0 : Software trigger<br>1 : $\overline{\text{AD}}_{\text{TRG}}$ trigger | RW |
| ADST | A/D conversion start flag | 0 : A/D conversion disabled<br>1 : A/D conversion started | RW |
| CKS0 | Frequency select bit 0 | See Note 2 for the ADCON2 register | RW |

Note 1: If the ADCON0 register is rewritten during A/D conversion, the conversion result will be indeterminate.
Note 2: After rewriting the MD1 to MD0 bits, set the CH2 to CH0 bits over again using another instruction.

A/D control register 1 (Note)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | 1 | | 0 | 0 | | |

| Symbol | Address | After reset |
|---|---|---|
| ADCON1 | $03D7_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| SCAN0 | A/D sweep pin select bit | Invalid in one-shot mode | RW |
| SCAN1 | | | RW |
| MD2 | A/D operation mode select bit 1 | Set to “0” when one-shot mode is selected | RW |
| $\overline{\text{(b3)}}$ | Reserved bit | Must always be set to “0” | RW |
| CKS1 | Frequency select bit1 | See Note 2 for the ADCON2 register | RW |
| VCUT | Vref connect bit (Note 2) | 1 : Vref connected | RW |
| OPA0 | External op-amp connection mode bit | b7 b6<br>0 0 : ANEX0 and ANEX1 are not used<br>0 1 : ANEX0 input is A/D converted<br>1 0 : ANEX1 input is A/D converted<br>1 1 : External op-amp connection mode | RW |
| OPA1 | | | RW |

Note 1: If the ADCON1 register is rewritten during A/D conversion, the conversion result will be indeterminate.
Note 2: If the VCUT bit is reset from “0” (Vref unconnected) to “1” (Vref connected), wait for 1 µs or more before starting A/D conversion.

**Figure 12.4 ADCON0 Register and ADCON1 Register (One-shot Mode)**

## 12.2 Repeat mode

In this mode, the input voltage on one selected pin is A/D converted repeatedly. Table 12.3 shows the specifications of repeat mode. Figure 12.5 shows the ADCON0 to ADCON1 registers in repeat mode.

**Table 12.3 Repeat Mode Specifications**

| Item | Specification |
|---|---|
| Function | The input voltage on one pin selected by the ADCON0 register's CH2 to CH0 bits and the ADCON1 register's OPA1 to OPA0 bits is A/D converted repeatedly. |
| A/D conversion start condition | • When the ADCON0 register's TRG bit is “0” (software trigger)<br>The ADCON0 register's ADST bit is set to “1” (A/D conversion starts)<br>• When the TRG bit is “1” ($\overline{\text{ADTRG}}$ trigger)<br>Input on the $\overline{\text{ADTRG}}$ pin changes state from high to low after the ADST bit is set to “1” (A/D conversion starts) |
| A/D conversion stop condition | Set the ADST bit to “0” (A/D conversion halted) |
| Interrupt request generation timing | None generated |
| Analog input pin | Select one pin from $AN_0$ to $AN_7$, ANEX0 to ANEX1 |
| Reading of result of A/D converter | Read one of the AD0 to AD7 registers that corresponds to the selected pin |

A/D control register 0 (Note 1)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | 0 | 1 | | | |

Symbol ADCON0 — Address 03D6₁₆ — After reset 00000XXX₂

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| CH0 | Analog input pin select bit | b2 b1 b0<br>0 0 0 : AN0 is selected<br>0 0 1 : AN1 is selected<br>0 1 0 : AN2 is selected<br>0 1 1 : AN3 is selected<br>1 0 0 : AN4 is selected<br>1 0 1 : AN5 is selected<br>1 1 0 : AN6 is selected<br>1 1 1 : AN7 is selected (Note 2) | RW |
| CH1 | | | RW |
| CH2 | | | RW |
| MD0 | A/D operation mode select bit 0 | b4 b3<br>0 1 : Repeat mode (Note 2) | RW |
| MD1 | | | RW |
| TRG | Trigger select bit | 0 : Software trigger<br>1 : ADTRG trigger | RW |
| ADST | A/D conversion start flag | 0 : A/D conversion disabled<br>1 : A/D conversion started | RW |
| CKS0 | Frequency select bit 0 | See Note 2 for the ADCON2 register | RW |

Note 1: If the ADCON0 register is rewritten during A/D conversion, the conversion result will be indeterminate.
Note 2: After rewriting the MD1 to MD0 bits, set the CH2 to CH0 bits over again using another instruction.

A/D control register 1 (Note)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | 1 | | 0 | 0 | | |

Symbol ADCON1 — Address 03D7₁₆ — After reset 00₁₆

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| SCAN0 | A/D sweep pin select bit | Invalid in repeat mode | RW |
| SCAN1 | | | RW |
| MD2 | A/D operation mode select bit 1 | Set to “0” when this mode is selected | RW |
| $\overline{\text{(b3)}}$ | Reserved bit | Must always be set to “0” | RW |
| CKS1 | Frequency select bit 1 | See Note 2 for the ADCON2 register | RW |
| VCUT | Vref connect bit (Note 2) | 1 : Vref connected | RW |
| OPA0 | External op-amp connection mode bit | b7 b6<br>0 0 : ANEX0 and ANEX1 are not used<br>0 1 : ANEX0 input is A/D converted<br>1 0 : ANEX1 input is A/D converted<br>1 1 : External op-amp connection mode | RW |
| OPA1 | | | RW |

Note 1: If the ADCON1 register is rewritten during A/D conversion, the conversion result will be indeterminate.
Note 2: If the VCUT bit is reset from “0” (Vref unconnected) to “1” (Vref connected), wait for 1 µs or more before starting A/D conversion.

**Figure 12.5 ADCON0 Register and ADCON1 Register (Repeat Mode)**

## 12.3 Single Sweep Mode

In this mode, the input voltages on selected pins are A/D converted, one pin at a time. Table 12.4 shows the specifications of single sweep mode. Figure 12.6 shows the ADCON0 to ADCON1 registers in single sweep mode.

**Table 12.4 Single Sweep Mode Specifications**

| Item | Specification |
|---|---|
| Function | The input voltages on pins selected by the ADCON1 register's SCAN1 to SCAN0 bits are A/D converted, one pin at a time. |
| A/D conversion start condition | • When the ADCON0 register's TRG bit is “0” (software trigger)<br>The ADCON0 register's ADST bit is set to “1” (A/D conversion starts)<br>• When the TRG bit is “1” ($\overline{\text{ADTRG}}$ trigger)<br>Input on the $\overline{\text{ADTRG}}$ pin changes state from high to low after the ADST bit is set to “1” (A/D conversion starts) |
| A/D conversion stop condition | • Completion of A/D conversion (If a software trigger is selected, the ADST bit is cleared to “0” (A/D conversion halted).)<br>• Set the ADST bit to “0” |
| Interrupt request generation timing | Completion of A/D conversion |
| Analog input pin | Select from $AN_0$ to $AN_1$ (2 pins), $AN_0$ to $AN_3$ (4 pins), $AN_0$ to $AN_5$ (6 pins), $AN_0$ to $AN_7$ (8 pins) |
| Reading of result of A/D converter | Read one of the AD0 to AD7 registers that corresponds to the selected pin |

A/D control register 0 (Note)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | 1 | 0 | | | |

| Symbol | Address | After reset |
|---|---|---|
| ADCON0 | $03D6_{16}$ | $00000XXX_2$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| CH0 | Analog input pin select bit | Invalid in single sweep mode | RW |
| CH1 | | | RW |
| CH2 | | | RW |
| MD0 | A/D operation mode select bit 0 | b4 b3<br>1 0 : Single sweep mode | RW |
| MD1 | | | RW |
| TRG | Trigger select bit | 0 : Software trigger<br>1 : ADTRG trigger | RW |
| ADST | A/D conversion start flag | 0 : A/D conversion disabled<br>1 : A/D conversion started | RW |
| CKS0 | Frequency select bit 0 | See Note 2 for the ADCON2 register | RW |

Note: If the ADCON0 register is rewritten during A/D conversion, the conversion result will be indeterminate.

A/D control register 1 (Note 1)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | 1 | | 0 | 0 | | |

| Symbol | Address | After reset |
|---|---|---|
| ADCON1 | $03D7_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| SCAN0 | A/D sweep pin select bit | When single sweep mode is selected<br>b1 b0<br>0 0 : $AN_0$ to $AN_1$ (2 pins)<br>0 1 : $AN_0$ to $AN_3$ (4 pins)<br>1 0 : $AN_0$ to $AN_5$ (6 pins)<br>1 1 : $AN_0$ to $AN_7$ (8 pins) | RW |
| SCAN1 | | | RW |
| MD2 | A/D operation mode select bit 1 | Set to “0” when single sweep mode is selected | RW |
| $\overline{\text{(b3)}}$ | Reserved bit | Must always be set to “0” | RW |
| CKS1 | Frequency select bit 1 | See Note 2 for the ADCON2 register | RW |
| VCUT | Vref connect bit (Note 2) | 1 : Vref connected | RW |
| OPA0 | External op-amp connection mode bit | b7 b6<br>0 0 : ANEX0 and ANEX1 are not used<br>0 1 : Must not be set<br>1 0 : Must not be set<br>1 1 : External op-amp connection mode | RW |
| OPA1 | | | RW |

Note 1: If the ADCON1 register is rewritten during A/D conversion, the conversion result will be indeterminate.
Note 2: If the VCUT bit is reset from “0” (Vref unconnected) to “1” (Vref connected), wait for 1 µs or more before starting A/D conversion.

**Figure 12.6 ADCON0 Register and ADCON1 Register (Single Sweep Mode)**

## 12.4 Repeat Sweep Mode 0

In this mode, the input voltages on selected pins are A/D converted repeatedly. Table 12.5 shows the specifications of repeat sweep mode 0. Figure 12.7 shows the ADCON0 to ADCON1 registers in repeat sweep mode 0.

**Table 12.5 Repeat Sweep Mode 0 Specifications**

| Item | Specification |
|---|---|
| Function | The input voltages on pins selected by the ADCON1 register's SCAN1 to SCAN0 bits are A/D converted repeatedly. |
| A/D conversion start condition | • When the ADCON0 register's TRG bit is “0” (software trigger)<br>The ADCON0 register's ADST bit is set to “1” (A/D conversion starts)<br>• When the TRG bit is “1” ($\overline{\text{ADTRG}}$ trigger)<br>Input on the $\overline{\text{ADTRG}}$ pin changes state from high to low after the ADST bit is set to “1” (A/D conversion starts) |
| A/D conversion stop condition | Set the ADST bit to “0” (A/D conversion halted) |
| Interrupt request generation timing | None generated |
| Analog input pin | Select from $AN_0$ to $AN_1$ (2 pins), $AN_0$ to $AN_3$ (4 pins), $AN_0$ to $AN_5$ (6 pins), $AN_0$ to $AN_7$ (8 pins) |
| Reading of result of A/D converter | Read one of the AD0 to AD7 registers that corresponds to the selected pin |

A/D control register 0 (Note)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | 1 | 1 | | | |

| Symbol | Address | After reset |
|---|---|---|
| ADCON0 | $03D6_{16}$ | $00000XXX_2$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| CH0 | Analog input pin select bit | Invalid in repeat sweep mode 0 | RW |
| CH1 | | | RW |
| CH2 | | | RW |
| MD0 | A/D operation mode select bit 0 | b4 b3<br>1 1 : Repeat sweep mode 0 or Repeat sweep mode 1 | RW |
| MD1 | | | RW |
| TRG | Trigger select bit | 0 : Software trigger<br>1 : $AD_{TRG}$ trigger | RW |
| ADST | A/D conversion start flag | 0 : A/D conversion disabled<br>1 : A/D conversion started | RW |
| CKS0 | Frequency select bit 0 | See Note 2 for the ADCON2 register | RW |

Note: If the ADCON0 register is rewritten during A/D conversion, the conversion result will be indeterminate.

A/D control register 1 (Note 1)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | 1 | | 0 | 0 | | |

| Symbol | Address | After reset |
|---|---|---|
| ADCON1 | $03D7_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| SCAN0 | A/D sweep pin select bit | When repeat sweep mode 0 is selected<br>b1 b0<br>0 0 : $AN_0$, $AN_1$ (2 pins)<br>0 1 : $AN_0$ to $AN_3$ (4 pins)<br>1 0 : $AN_0$ to $AN_5$ (6 pins)<br>1 1 : $AN_0$ to $AN_7$ (8 pins) | RW |
| SCAN1 | | | RW |
| MD2 | A/D operation mode select bit 1 | Set to “0” when repeat sweep mode 0 is selected | RW |
| $\overline{\text{(b3)}}$ | Reserved bit | Must always be set to “0” | RW |
| CKS1 | Frequency select bit 1 | See Note 2 for the ADCON2 register | RW |
| VCUT | Vref connect bit (Note 2) | 1 : Vref connected | RW |
| OPA0 | External op-amp connection mode bit | b7 b6<br>0 0 : ANEX0 and ANEX1 are not used<br>0 1 : Must not be set<br>1 0 : Must not be set<br>1 1 : External op-amp connection mode | RW |
| OPA1 | | | RW |

Note 1: If the ADCON1 register is rewritten during A/D conversion, the conversion result will be indeterminate.
Note 2: If the VCUT bit is reset from “0” (Vref unconnected) to “1” (Vref connected), wait for 1 µs or more before starting A/D conversion.

**Figure 12.7 ADCON0 Register and ADCON1 Registers (Repeat Sweep Mode 0)**

## 12.5 Repeat Sweep Mode 1

In this mode, the input voltages on all pins are A/D converted repeatedly, with priority given to the selected pins. Table 12.6 shows the specifications of repeat sweep mode 1. Figure 12.8 shows the ADCON0 to ADCON1 registers in repeat sweep mode 1.

**Table 12.6 Repeat Sweep Mode 1 Specifications**

| Item | Specification |
|---|---|
| Function | The input voltages on all selected pins are A/D converted repeatedly,with priority given to pins selected by the ADCON1 register's SCAN1 to SCAN0 bits.<br>Example : If $AN_0$ selected, input voltages are A/D converted in order of<br>$AN_0 \rightarrow AN_1 \rightarrow AN_0 \rightarrow AN_2 \rightarrow AN_0 \rightarrow AN_3$, and so on. |
| A/D conversion start condition | • When the ADCON0 register's TRG bit is “0” (software trigger)<br>The ADCON0 register's ADST bit is set to “1” (A/D conversion starts)<br>• When the TRG bit is “1” ($\overline{ADTRG}$ trigger)<br>Input on the $\overline{ADTRG}$ pin changes state from high to low after the ADST bit is set to “1” (A/D conversion starts) |
| A/D conversion stop condition | Set the ADST bit to “0” (A/D conversion halted) |
| Interrupt request generation timing | None generated |
| Analog input pins to be given priority when A/D converted | Select from $AN_0$ (1 pins), $AN_0$ to $AN_1$ (2 pins), $AN_0$ to $AN_2$ (3 pins), $AN_0$ to $AN_3$ (4 pins) |
| Reading of result of A/D converter | Read one of the AD0 to AD7 registers that corresponds to the selected pin |

A/D control register 0 (Note)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | 1 | 1 | | | |

Symbol ADCON0 | Address 03D6$_{16}$ | After reset 00000XXX$_2$

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| CH0 | Analog input pin select bit | Invalid in repeat sweep mode 1 | RW |
| CH1 | | | RW |
| CH2 | | | RW |
| MD0 | A/D operation mode select bit 0 | b4 b3<br>1 1 : Repeat sweep mode 0 or Repeat sweep mode 1 | RW |
| MD1 | | | RW |
| TRG | Trigger select bit | 0 : Software trigger<br>1 : ADTRG trigger | RW |
| ADST | A/D conversion start flag | 0 : A/D conversion disabled<br>1 : A/D conversion started | RW |
| CKS0 | Frequency select bit 0 | See Note 2 for the ADCON2 register | RW |

Note: If the ADCON0 register is rewritten during A/D conversion, the conversion result will be indeterminate.

A/D control register 1 (Note 1)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | 1 | | 0 | 1 | | |

Symbol ADCON1 | Address 03D7$_{16}$ | After reset 00$_{16}$

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| SCAN0 | A/D sweep pin select bit | When repeat sweep mode 1 is selected<br>b1 b0<br>0 0 : AN$_0$ (1 pin)<br>0 1 : AN$_0$, AN$_1$ (2 pins)<br>1 0 : AN$_0$ to AN$_2$ (3 pins)<br>1 1 : AN$_0$ to AN$_3$ (4 pins) | RW |
| SCAN1 | | | RW |
| MD2 | A/D operation mode select bit 1 | Set to “1” when repeat sweep mode 1 is selected | RW |
| $\overline{\text{(b3)}}$ | Reserved bit | Must always be set to “0” | RW |
| CKS1 | Frequency select bit 1 | See Note 2 for the ADCON2 register | RW |
| VCUT | Vref connect bit (Note 2) | 1 : Vref connected | RW |
| OPA0 | External op-amp connection mode bit | b7 b6<br>0 0 : ANEX0 and ANEX1 are not used<br>0 1 : Must not be set<br>1 0 : Must not be set<br>1 1 : External op-amp connection mode | RW |
| OPA1 | | | RW |

Note 1: If the ADCON1 register is rewritten during A/D conversion, the conversion result will be indeterminate.
Note 2: If the VCUT bit is reset from “0” (Vref unconnected) to “1” (Vref connected), wait for 1 µs or more before starting A/D conversion.

**Figure 12.8 ADCON0 Register and ADCON1 Register (Repeat Sweep Mode 1)**

## 12.6 Sample and Hold

If the ADCON2 register's SMP bit is set to "1" (with sample-and-hold), the conversion speed per pin is increased to 28 $\phi_{AD}$ cycles for 8-bit resolution. Sample-and-hold is effective in all operation modes.
Select whether or not to use the sample-and-hold function before starting A/D conversion.

## 12.7 Extended Analog Input Pins

In one-shot and repeat modes, the ANEX0 and ANEX1 pins can be used as analog input pins. Use the ADCON1 register's OPA1 to OPA0 bits to select whether or not use ANEX0 and ANEX1.
The A/D conversion results of ANEX0 and ANEX1 inputs are stored in the AD0 and AD1 registers, respectively.

## 12.8 External Operation Amp Connection Mode

Multiple analog inputs can be amplified using a single external op-amp via the ANXE0 and ANEX1 pins.
Set the ADCON1 register's OPA1 OPA0 bits to '$11_2$' (external op-amp connection mode). The inputs from ANi (i = 0 to 7) are output from the ANEX0 pin. Amplify this output with an external op-amp before sending it back to the ANEX1 pin. The A/D conversion result is stored in the corresponding ADi register. The A/D conversion speed depends on the response characteristics of the external op-amp. Note that the ANXE0 and ANEX1 pins cannot be directly connected to each other. Figure 12.9 is an example of how to connect the pins in external operation amp.

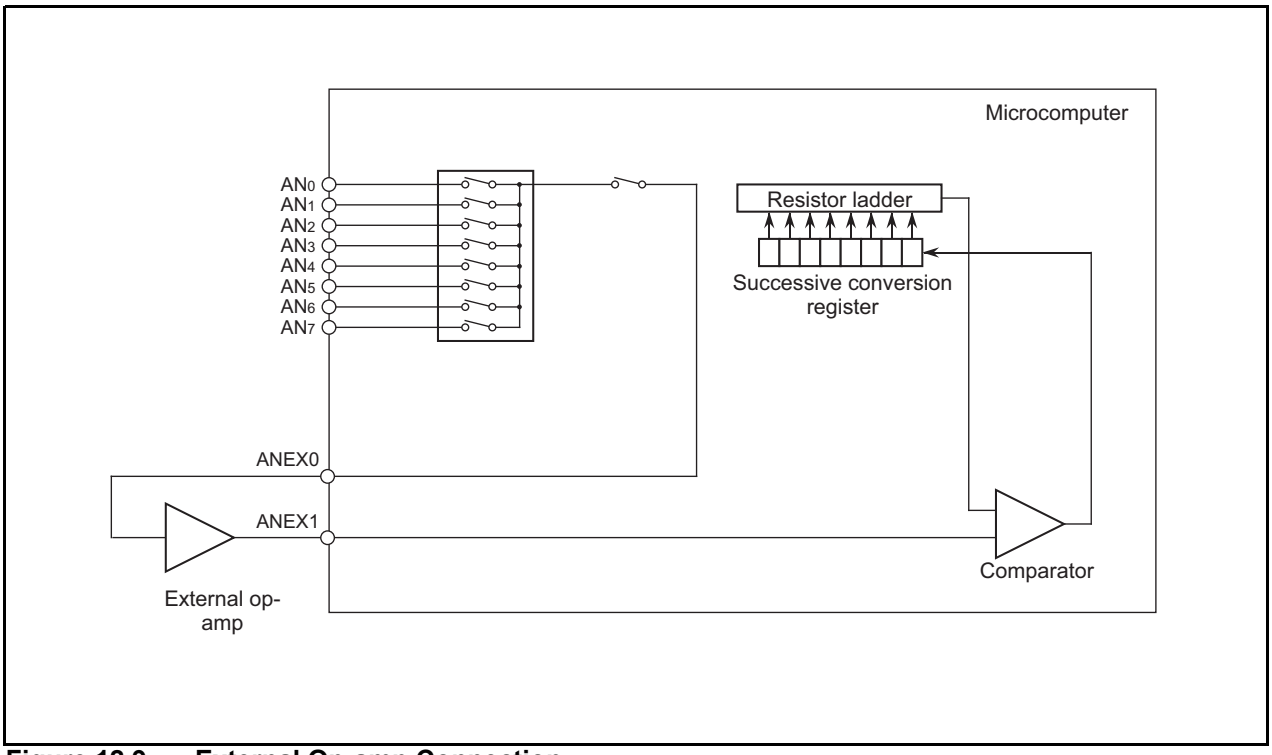



**Figure 12.9 External Op-amp Connection**

## 12.9 Current Consumption Reducing Function

When not using the A/D converter, its resistor ladder and reference voltage input pin (VREF) can be separated using the ADCON1 register's VCUT bit. When separated, no current will flow from the VREF pin into the resistor ladder, helping to reduce the power consumption of the chip.

To use the A/D converter, set the VCUT bit to "1" (VREF connected) and then set the ADCON0 register's ADST bit to "1" (A/D conversion start). The VCUT and ADST bits cannot be set to "1" at the same time.

Nor can the VCUT bit be set to "0" (VREF unconnected) during A/D conversion.

## 12.10 Analog Input Pin and External Sensor Equivalent Circuit Example

Figure 12.10 shows analog input pin and external sensor equivalent circuit example.

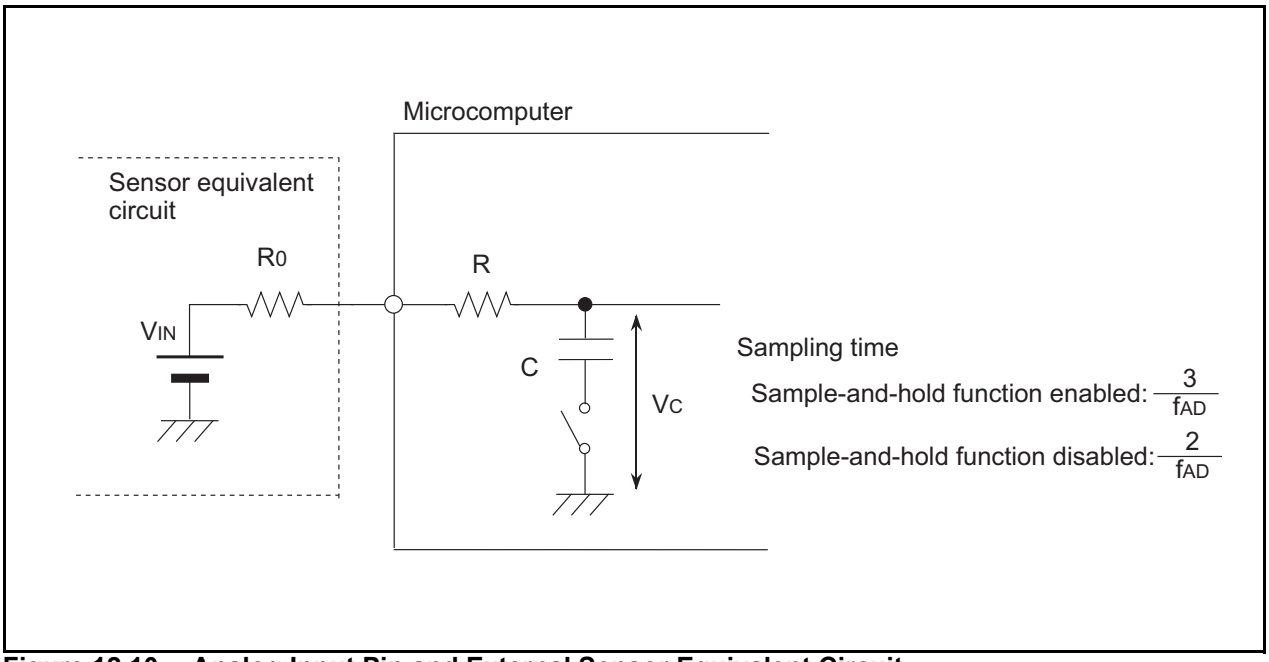



**Figure 12.10 Analog Input Pin and External Sensor Equivalent Circuit**

## 12.11 Caution of Using A/D Converter

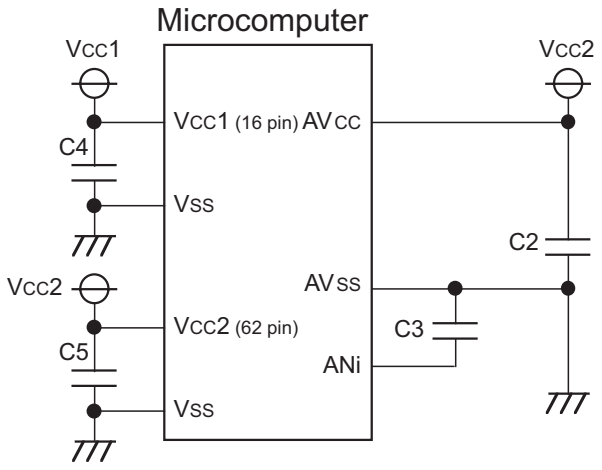
(1) Make sure the port direction bits for those pins that are used as analog inputs are set to "0" (input mode). Also, if the ADCON0 register's TGR bit = 1 (external trigger), make sure the port direction bit for the ADTRG pin is set to "0" (input mode).

(2) To prevent noise-induced device malfunction or latchup, as well as to reduce conversion errors, insert capacitors between the AVCC, VREF, and analog input pins (ANi (i=0 to 7)) each and the AVSS pin. Similarly, insert a capacitor between the VCC pin and the VSS pin. Figure 12.11 is an example connection of each pin.

(3) If the CPU reads the ADi register (i = 0 to 7) at the same time the conversion result is stored in the ADi register after completion of A/D conversion, an incorrect value may be stored in the ADi register. This problem occurs when a divide-by-n clock derived from the main clock or a subclock is selected for CPU clock.

- When operating in one-shot or single-sweep mode
  Check to see that A/D conversion is completed before reading the target ADi register. (Check the IR bit in the ADIC register to see if A/D conversion is completed.)
- When operating in repeat mode or repeat sweep mode 0 or 1
  Use the main clock for CPU clock directly without dividing it.

(4) If A/D conversion is forcibly terminated while in progress by setting the ADCON0 register's ADST bit to "0" (A/D conversion halted), the conversion result of the A/D converter is indeterminate. The contents of ADi registers irrelevant to A/D conversion may also become indeterminate. If while A/D conversion is underway the ADST bit is cleared to "0" in a program, ignore the values of all ADi registers.




ANi: ANi (i=0 to 7)

Note 1: C1≥0.47μF, C2≥0.47μF, C3≥100pF, C4≥0.1μF, C5≥0.1μF (reference)
Note 2: Use thick and shortest possible wiring to connect capacitors.

**Figure 12.11 VCC, VSS, AVCC, AVSS, VREF and ANi Connection**

# 13. CRC Calculation

The Cyclic Redundancy Check (CRC) operation detects an error in data blocks. The microcomputer uses a generator polynomial of CRC_CCITT ($X^{16} + X^{12} + X^5 + 1$) to generate CRC code.
The CRC code consists of 16 bits which are generated for each data block in given length, separated in 8 bit units. After the initial value is set in the CRCD register, the CRC code is set in that register each time one byte of data is written to the CRCIN register. CRC code generation for one-byte data is finished in two cycles.
Figure 13.1 shows the block diagram of the CRC circuit. Figure 13.2 shows the CRC-related registers.
Figure 13.3 shows the calculation example using the CRC operation.

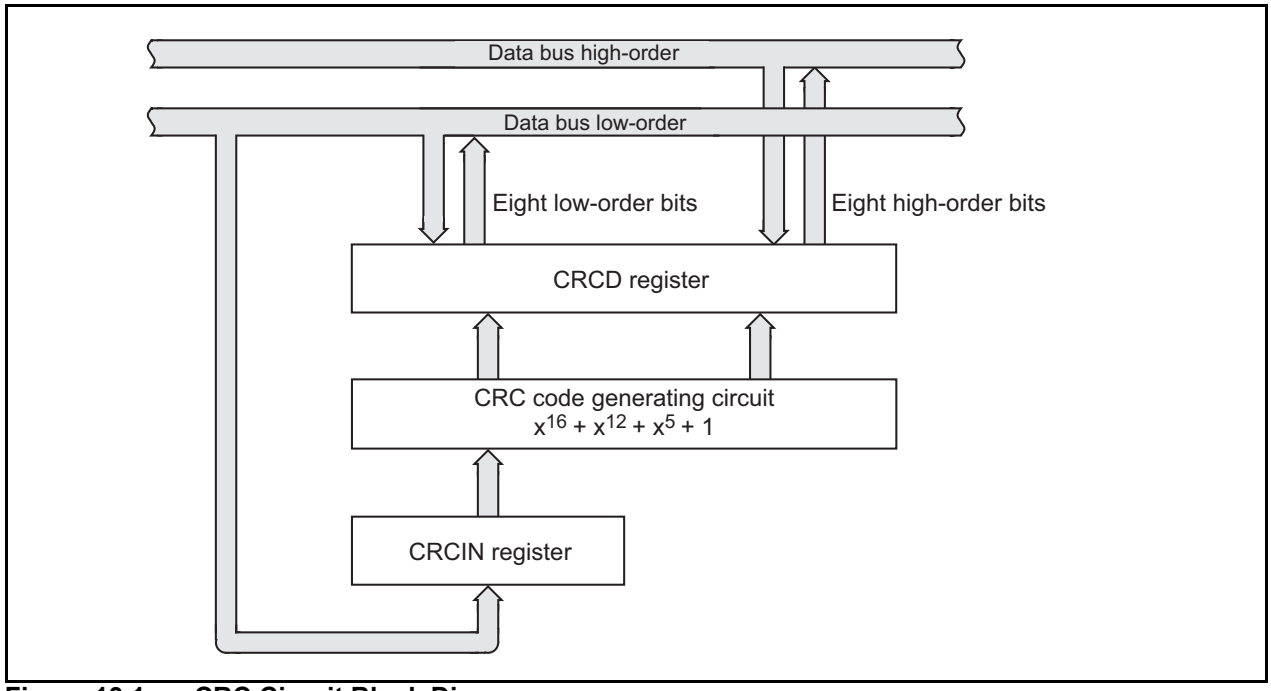



**Figure 13.1 CRC Circuit Block Diagram**

CRC data register

(b15) (b8) b7 b0 b7 b0

| Symbol | Address | After reset |
|---|---|---|
| CRCD | 03BD16 to 03BC16 | Indeterminate |

| Function | Setting range | RW |
|---|---|---|
| When data is written to the CRCIN register after setting the initial value in the CRCD register, the CRC code can be read out from the CRCD register. | 000016 to FFFF16 | RW |

CRC input register

b7 b0

| Symbol | Address | After reset |
|---|---|---|
| CRCIN | 03BE16 | Indeterminate |

| Function | Setting range | RW |
|---|---|---|
| Data input | 0016 to FF16 | RW |

**Figure 13.2 CRCD Register and CRCIN Register**

**Setup procedure and CRC operation when generating CRC code "80C4$_{16}$"**

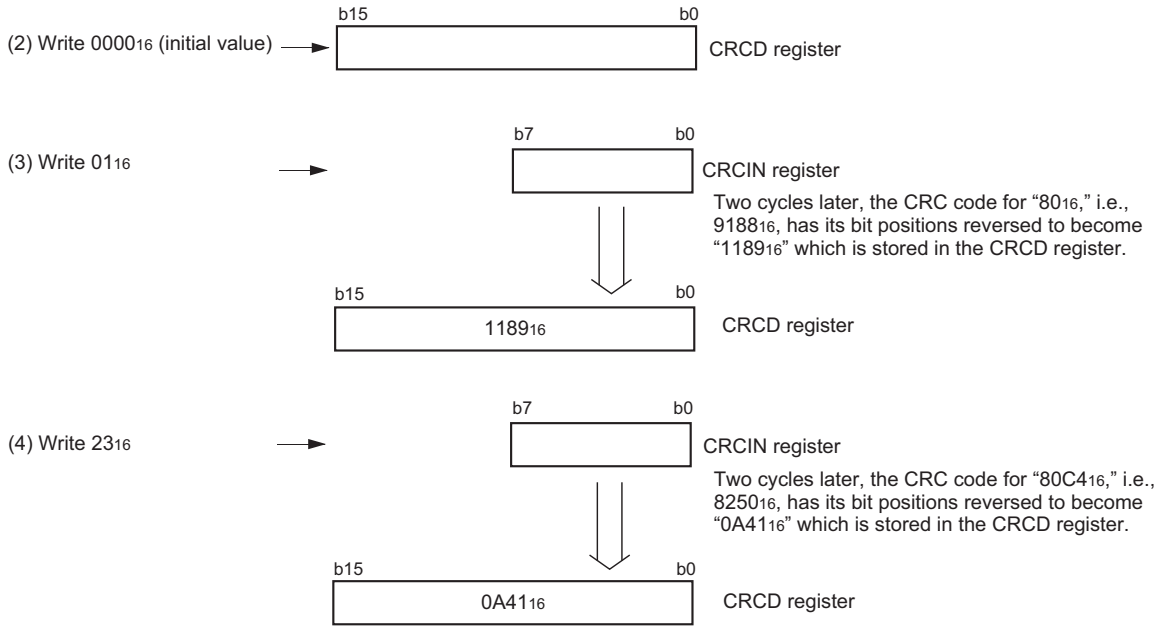
(a) CRC operation performed by the M16C

CRC code: Remainder of a division in which the value written to the CRCIN register with its bit positions reversed is divided by the generator polynomial

Generator polynomial: $X^{16} + X^{12} + X^5 + 1$ ($1\ 0001\ 0000\ 0010\ 0001_2$)

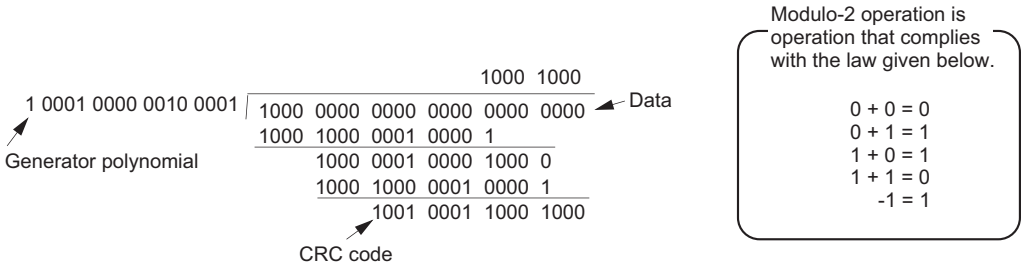
(b) Setting procedure

(1) Reverse the bit positions of the value "80C4$_{16}$" bytewise in a program.
"80$_{16}$" → "01$_{16}$", "C4$_{16}$" → "23$_{16}$"




(c) Details of CRC operation

In the case of (3) above, the value written to the CRCIN register "01$_{16}$ ($00000001_2$)" has its bit positions reversed to become "$10000000_2$." The value "1000 0000 0000 0000 0000 $0000_2$" derived from that by adding 16 digits and the CRCD register's initial value "0000$_{16}$" are added, the result of which is divided by the generator polynomial using modulo-2 arithmetic.




The value "0001 0001 1000 $1001_2$ (1189$_{16}$)" derived from the remainder "1001 0001 1000 $1000_2$ (9188$_{16}$)" by reversing its bit positions may be read from the CRCD register.

If operation (4) above is performed subsequently, the value written to the CRCIN register "23$_{16}$ ($00100011_2$)" has its bit positions reversed to become "$11000100_2$. The value "1100 0100 0000 0000 0000 $0000_2$" derived from that by adding 16 digits and the remainder in (3) "1001 0001 1000 $1000_2$" which is left in the CRCD register are added, the result of which is divided by the generator polynomial using modulo-2 arithmetic.
The value "0000 1010 0100 $0001_2$ (0A41$_{16}$)" derived from the remainder by reversing its bit positions may be read from the CRCD register.

**Figure 13.3 CRC Calculation**

# 14. Expansion Function

## 14.1 Expansion function description

Expansion function consists of CRC operation function, data slice function and humming decoder function. Each function is controlled by expansion memories.

1. CRC operation function
   It performs error detection of a code, and error correction.
2. Data slice function
   It performs data acquisition to get such format data as below.
   Hardware : TELETEXT, PDC, VPS, VBI and EPG-J
   Software : WSS, CC, CC2X and ID-1
3. Humming decoder function
   It performs 8/4 humming and 24/18 humming

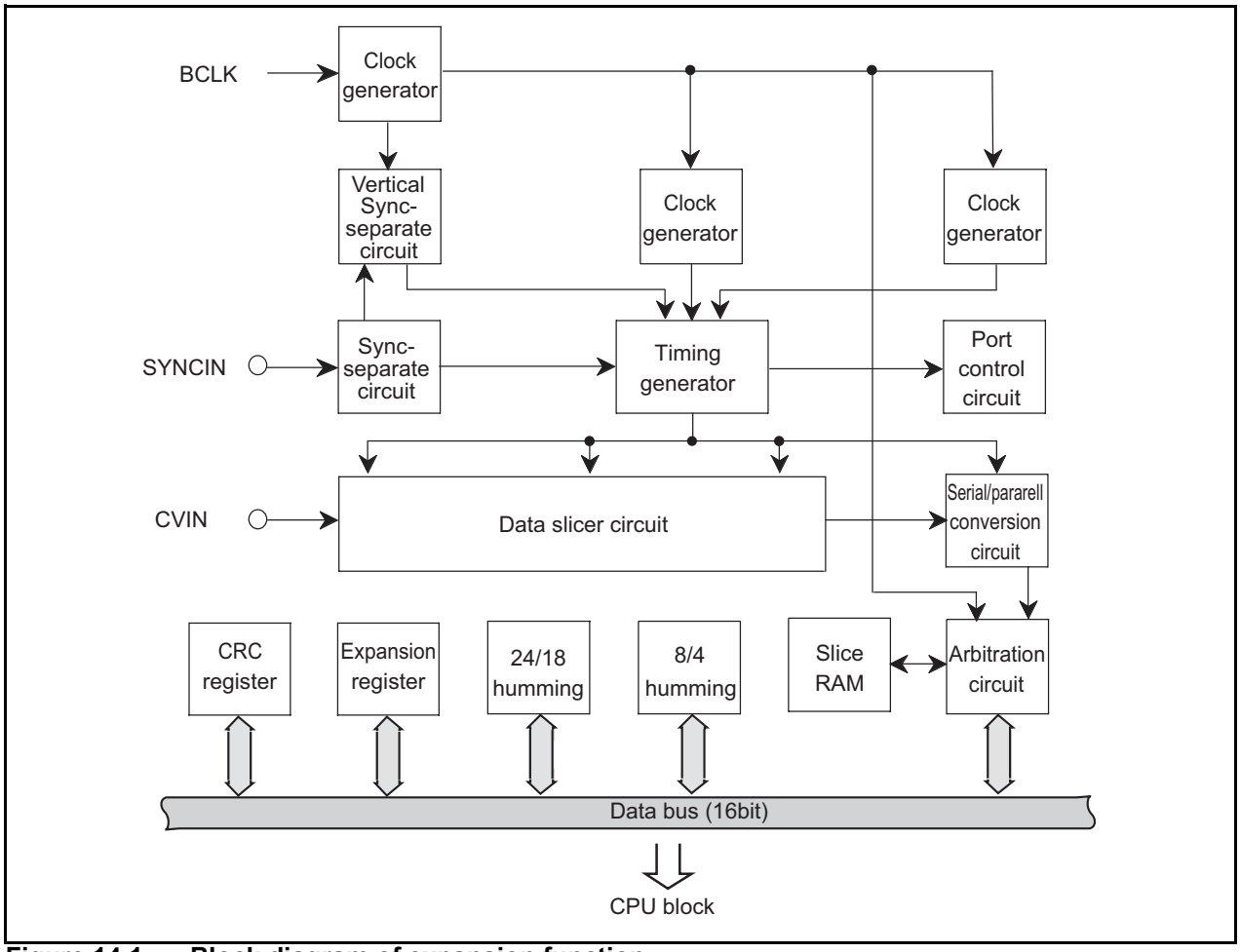



**Figure 14.1 Block diagram of expansion function**

## 14.2 Expansion memory

Expansion function memory is divided by 3 patterns ; Slice RAM, CRC registers and expansion registers (Humming decoder operates by the register placed on SFR). Data writing and read out to the Slice RAM, CRC registers and the expansion registers are carried out per 16 bit unit by the data setting register (addresses $020E_{16}$, $0210_{16}$, $0212_{16}$, $0214_{16}$, $0216_{16}$ and $0218_{16}$) placed on SFR.
Contents of each memory and data setting register are shown in Table 14.1.

**Table 14.1 Expansion memory composition**

| Expansion memory | Contents | Data setting register |
|---|---|---|
| Slice RAM | This register holds acquired data. | Slice RAM address control register ($020E_{16}$)<br>Slice RAM data control register ($0210_{16}$) |
| CRC register | This register controls a set up generation polynomial and code data. | CRC register address control register ($0212_{16}$)<br>CRC register data control register ($0214_{16}$) |
| Expansion register | This register performs data slicer control and VBI encoder control. | Expansion register address control register ($0216_{16}$)<br>Expansion register data control register ($0218_{16}$) |

## 14.3 Slice RAM

Slice RAM stores 18-line slice data. There are several types of Slice data : PDC, VPS, VBI, XDS, WSS, etc. All data are stored to addresses which corresponds to slice line (ex. 22 line' data is stored to addresses $200_{16}$ to $217_{16}$ ). 24 addresses (SR00x to SR17x) are prepared for 1 line, slice data is stored in order from LSB side. Then, slice data type and field information are stored to the top address of each line.
Slice RAM composition is shown in Table 14.2.

**Table 14.2 Slice RAM composition**

| Slice RAM addresses (SA9 to SA0) | SD15 | SD14 | SD13 | SD12 | SD11 | SD10 | SD9 | SD8 | SD7 | SD6 | SD5 | SD4 | SD3 | SD2 | SD1 | SD0 | Remarks (Note1) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $000_{16}$<br>$001_{16}$<br>to<br>$016_{16}$<br>$017_{16}$ | SR00F<br>SR01F<br>to<br>SR16F<br>SR17F | SR00E<br>SR01E<br>to<br>SR16E<br>SR17E | SR00D<br>SR01D<br>to<br>SR16D<br>SR17D | SR00C<br>SR01C<br>to<br>SR16C<br>SR17C | SR00B<br>SR01B<br>to<br>SR16B<br>SR17B | SR00A<br>SR01A<br>to<br>SR16A<br>SR17A | SR009<br>SR019<br>to<br>SR169<br>SR179 | SR008<br>SR018<br>to<br>SR168<br>SR178 | SR007<br>SR017<br>to<br>SR167<br>SR177 | SR006<br>SR016<br>to<br>SR166<br>SR176 | SR005<br>SR015<br>to<br>SR165<br>SR175 | SR004<br>SR014<br>to<br>SR164<br>SR174 | SR003<br>SR013<br>to<br>SR163<br>SR173 | SR002<br>SR012<br>to<br>SR162<br>SR172 | SR001<br>SR011<br>to<br>SR161<br>SR171 | SR000<br>SR010<br>to<br>SR160<br>SR170 | 6th line or 318th line slice data |
| $018_{16}$<br>to<br>$01F_{16}$ | Unused area | | | | | | | | | | | | | | | | |
| $020_{16}$<br>to<br>$037_{16}$ | SR00F<br>to<br>SR17F | SR00E<br>to<br>SR17E | SR00D<br>to<br>SR17D | SR00C<br>to<br>SR17C | SR00B<br>to<br>SR17B | SR00A<br>to<br>SR17A | SR009<br>to<br>SR179 | SR008<br>to<br>SR178 | SR007<br>to<br>SR177 | SR006<br>to<br>SR176 | SR005<br>to<br>SR175 | SR004<br>to<br>SR174 | SR003<br>to<br>SR173 | SR002<br>to<br>SR172 | SR001<br>to<br>SR171 | SR000<br>to<br>SR170 | 7th line or 319th line slice data |
| $040_{16}$<br>to<br>$1F7_{16}$ | ⋮ | | | | | | | | | | | | | | | | 8th line to 21th line or 320th line to 333 line slice data |
| $200_{16}$<br>to<br>$217_{16}$ | SR00F<br>to<br>SR17F | SR00E<br>to<br>SR17E | SR00D<br>to<br>SR17D | SR00C<br>to<br>SR17C | SR00B<br>to<br>SR17B | SR00A<br>to<br>SR17A | SR009<br>to<br>SR179 | SR008<br>to<br>SR178 | SR007<br>to<br>SR177 | SR006<br>to<br>SR176 | SR005<br>to<br>SR175 | SR004<br>to<br>SR174 | SR003<br>to<br>SR173 | SR002<br>to<br>SR172 | SR001<br>to<br>SR171 | SR000<br>to<br>SR170 | 22th line or 334th line slice data |
| $220_{16}$<br>to<br>$237_{16}$ | SR00F<br>to<br>SR17F | SR00E<br>to<br>SR17E | SR00D<br>to<br>SR17D | SR00C<br>to<br>SR17C | SR00B<br>to<br>SR17B | SR00A<br>to<br>SR17A | SR009<br>to<br>SR179 | SR008<br>to<br>SR178 | SR007<br>to<br>SR177 | SR006<br>to<br>SR176 | SR005<br>to<br>SR175 | SR004<br>to<br>SR174 | SR003<br>to<br>SR173 | SR002<br>to<br>SR172 | SR001<br>to<br>SR171 | SR000<br>to<br>SR170 | 23th line or 335th line slice data |

Note 1. This is the line to support when the PAL video signal is sliced and setting the expansion registers to VPS_VP8 to VPS_VP0 (bits 8 to 0 in address $29_{16}$) = "$4_{16}$".

For accessing to Slice RAM data, set accessing address (SA9 to SA0) (shown in Table 14.2) to Slice RAM address control register (address $020E_{16}$ ). Then read out data from Slice RAM data control register (address $0210_{16}$ ). When end the data reading, Slice RAM address control register increments address automatically. Then, next address data reading is possible. Do not access to unused area of each character codes. Must set address to each line because unused area has no address' automatically increment.
Slice RAM bit composition is shown in Figure 14.2, Slice RAM access registers are shown in Figure14.3 and Slice RAM access block diagram is shown in Figure 14.4

.

## Slice RAM bit composition

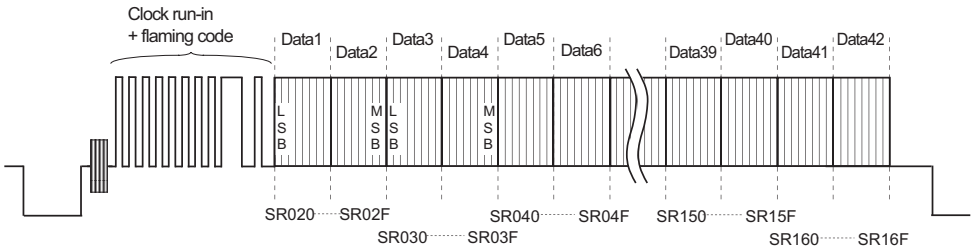
The each head address of the address is corresponded to slice line following slice information.

| | SR00F to SR004 | SR003 | SR002 | SR001 | SR000 |
|---|---|---|---|---|---|
| Line register 3 | 0 | field* | 0 | 1 | 1 |
| Line register 2 | 0 | field* | 0 | 1 | 0 |
| Line register 1 | 0 | field* | 0 | 0 | 1 |
| Other | 0 | field* | 0 | 0 | 0 |

* field
* the first field : 1
the second field : 0

(1) PDC

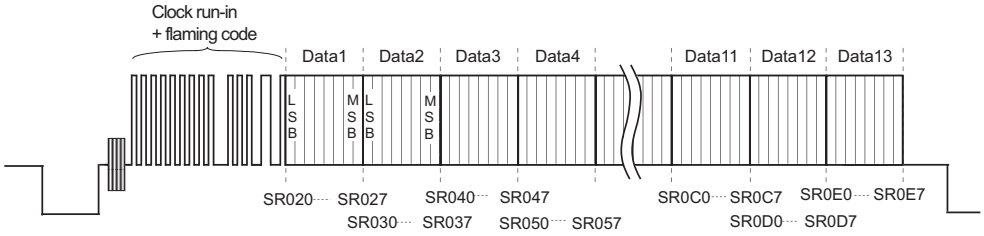
In case of the PDC data, 16 bits (2 data) are stored for the 1 address from the LSB side.




Note. The expansion register is the slice data storing pattern when setting the START (bit 1 in address $28_{16}$ bit) to "1". SR17x is the unused area.

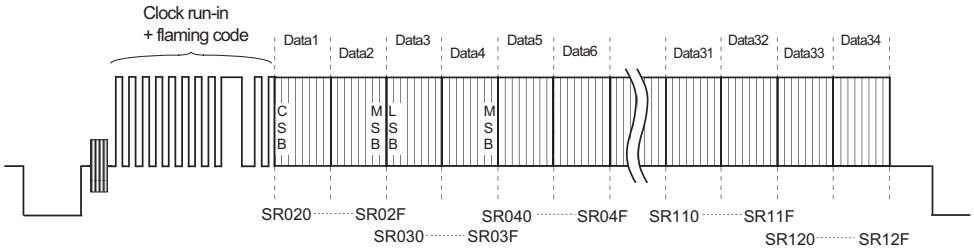
(2) VPS

In case of the VPS data or the VBI data, 8 bits (a data) are stored for an address from the LSB side.
Low-order 8 bits hold the slice data. And, high-order 8 bits hold warning bit, when the send data is not recognized as bi-phase type.
The case of bi-phase data ="1,0" or "0,1" (the bi-phase type) becomes "0" for this warning bit, and it becomes "1" in bi-phase data="0,0" or "1,1" (it is not the bi-phase type).
(For example, bi-phase data of SR011 is "0,0" or "1,1", "1" is set to SR019.)




Note. The expansion register is the slice data storing pattern when setting the START (bit 1 in address $28_{16}$ bit) to "1". From SR0Fx to SR17x are the unused area.

(3) EPG-J




Note. The expansion register is the slice data storing pattern when setting the START (bit 1 in address $28_{16}$ bit) to "1". From SR13x to SR17x are the unused area.

**Figure 14.2 Slice RAM bit composition**

Slice RAM address control register

b15 b9 b8 b7 b0

| Symbol | Address | When reset |
|---|---|---|
| SA | 020E16 | 000016 |

| Function | Setting possible value | R | W |
|---|---|---|---|
| Specify accessing Slice RAM address | 000016 to 23716 | X | O |
| Nothing is assigned.<br>When write, set to "0."<br>When read, its content is indeterminate. | | – | – |

Note 1 : When access to Slice RAM, Slice RAM address control register (020E16) should be set at first.
Slice RAM address control register increments by accessing Slice RAM data control register. So, it is not neccesary to setting the next Slice RAM address.

Note 2 : When read Slice RAM data by software during slicer operation, access to Slice RAM after 1 horizontal synchronous period from the completion of a SLICEON (refer to 14.6 Expansion Register Construction Composition for a SLICEON period).

Slice RAM data control register

b15 b9 b8 b7 b0

| Symbol | Address | When reset |
|---|---|---|
| SD | 021016 | 000016 |

| Function | R | W |
|---|---|---|
| Read out the data of Slice RAM.<br>Read out data of Slice RAM which is specified by Slice RAM address control register (address 020E16) by reading this register. | O | X |

Note : Data access must be 16-bit unit. 8-bit unit access is disable.

**Figure 14.3 Slice RAM access registers.**

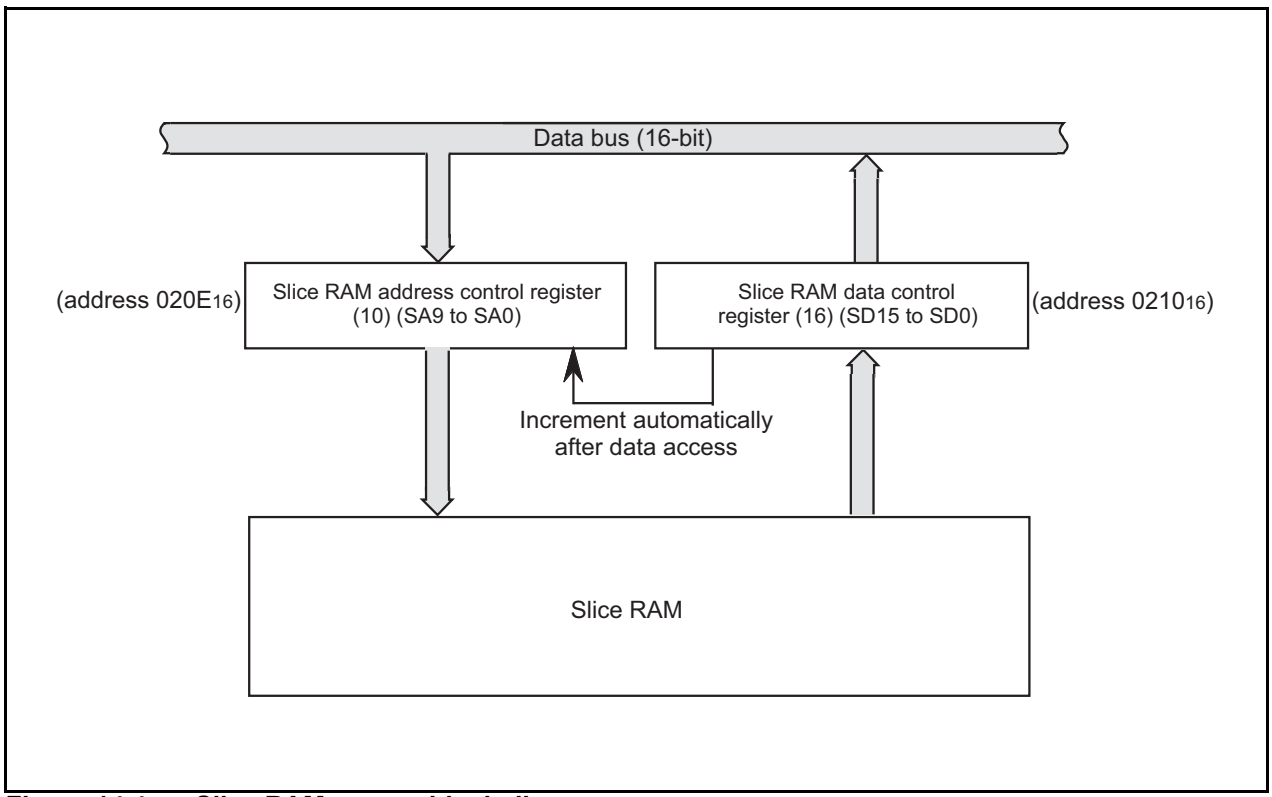



**Figure 14.4 Slice RAM access block diagram**

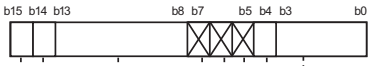
## 14.4 CRC Operation Circuit (EPG-J)

CRC operation circuit (EPG-J) is a circuit for performing error detection and error correction by the 272-190 shortening difference set cyclic code which is a coding system in a data multiplex broadcast.
CRC register consists of registers shown in Figure 14.5. CRC register can perform error detection and error correction by majority logic by setting up a generator polynomial, code data, etc. CRC register composition is shown in Table 14.3.

**Table 14.3 CRC register composition**

| CA3 to CA0 | CD15 | CD14 | CD13 | CD12 | CD11 | CD10 | CD9 | CD8 | CD7 | CD6 | CD5 | CD4 | CD3 | CD2 | CD1 | CD0 | Remarks |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0016 | DAOUT15 | DAOUT14 | DAOUT13 | DAOUT12 | DAOUT11 | DAOUT10 | DAOUT9 | DAOUT8 | DAOUT7 | DAOUT6 | DAOUT5 | DAOUT4 | DAOUT3 | DAOUT2 | DAOUT1 | DAOUT0 | |
| 0116 | _ | _ | _ | _ | _ | CRC_ERR10 | CRC_ERR09 | CRC_ERR08 | CRC_ERR07 | CRC_ERR06 | CRC_ERR05 | CRC_ERR04 | CRC_ERR03 | CRC_ERR02 | CRC_ERR01 | CRC_ERR00 | |
| 0216 | CRC_66 | CRC_67 | CRC_68 | CRC_69 | CRC_70 | CRC_71 | CRC_72 | CRC_73 | CRC_74 | CRC_75 | CRC_76 | CRC_77 | CRC_78 | CRC_79 | CRC_80 | CRC_81 | |
| 0316 | CRC_50 | CRC_51 | CRC_52 | CRC_53 | CRC_54 | CRC_55 | CRC_56 | CRC_57 | CRC_58 | CRC_59 | CRC_60 | CRC_61 | CRC_62 | CRC_63 | CRC_64 | CRC_65 | |
| 0416 | CRC_34 | CRC_35 | CRC_36 | CRC_37 | CRC_38 | CRC_39 | CRC_40 | CRC_41 | CRC_42 | CRC_43 | CRC_44 | CRC_45 | CRC_46 | CRC_47 | CRC_48 | CRC_49 | |
| 0516 | CRC_18 | CRC_19 | CRC_20 | CRC_21 | CRC_22 | CRC_23 | CRC_24 | CRC_25 | CRC_26 | CRC_27 | CRC_28 | CRC_29 | CRC_30 | CRC_31 | CRC_32 | CRC_33 | |
| 0616 | CRC_02 | CRC_03 | CRC_04 | CRC_05 | CRC_06 | CRC_07 | CRC_08 | CRC_09 | CRC_10 | CRC_11 | CRC_12 | CRC_13 | CRC_14 | CRC_15 | CRC_16 | CRC_17 | |
| 0716 | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | CRC_00 | CRC_01 | |
| 0816 | REG_C81 | REG_C80 | REG_C79 | REG_C78 | REG_C77 | REG_C76 | REG_C75 | REG_C74 | REG_C73 | REG_C72 | REG_C71 | REG_C70 | REG_C69 | REG_C68 | REG_C67 | REG_C66 | |
| 0916 | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | |
| 0A16 | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | |
| 0B16 | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | |
| 0C16 | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | |
| 0D16 | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | _ | CRC16SEL | _ | _ | _ | _ | |

CRC register address control register




| Symbol | address | at Reset |
|---|---|---|
| CA | 021216 | 000016 |

| Function | | The value which can be set up | R | W |
|---|---|---|---|---|
| Specify accessing CRC register address. | | 0016 to 0D16 | ○ | ○ |
| CRC register address automatic increment.<br>0: enable / 1 : disable (Notes 2) | | – | × | ○ |
| Nothing is assigned.<br>When write, set to "0." When read, its content is determinate. | | | – | – |
| CRCLOOP 0 to 5 | The number of times of a CRC operation repetition. | 0016 to 3F16 | ○ | ○ |
| CRCCHANGE | Error detection / error correction Selection setting<br>0:error detection mode / 1: Error correction mode | – | ○ | ○ |
| CRCON | CRC operation<br>0: Stop/1 : Operation (Note 3) | – | ○ | ○ |

Notes 1: When access to CRC register, must be set CRC register address at first, then use CRC register data control register (021416).
Notes 2: When bit 4 = "0" setting, CRC register data control register increments by accessing CRC register data control register, so it is not necessary to setting the next CRC register address. When bit 4 = "1" setting, the address is fixed.
Notes 3: When bit 15 = "0" setting, the value of a CRC data register (address (CA3 to CA0) =01 to 07) is cleared.

CRC register data control register




| Symbol | address | at Reset |
|---|---|---|
| CD | 021416 | 000016 |

| Function | The value which can be set up | R | W |
|---|---|---|---|
| Write and read out the data of CRC register which is specified by CRC register address control register (address 021216) | 000016 to FFFF16 | ○ | ○ |

Note: Data access must be 16-bit unit. 8-bit unit access is disable.

**Figure 14.5 Composition of CRC register access related register**

For accessing to CRC register data, set accessing address (CA3 to CA0) (shown in Table 14.3) to CRC register address control register (address 021216). Then write data (CD15 to CD0) by CRC register data control register (address 021416). When end the data accessing, CRC register address control register increments address automatically. Then, next address data writing is possible.
CRC register access registers are shown in Figure 14.5, CRC register access block diagram is shown in Figure 14.6. The operation example of CRC operation circuit is shown in Figure 14.7. The example of program is shown in Figure 14.8, and CRC register bit compositions are shown in p191 to p199.

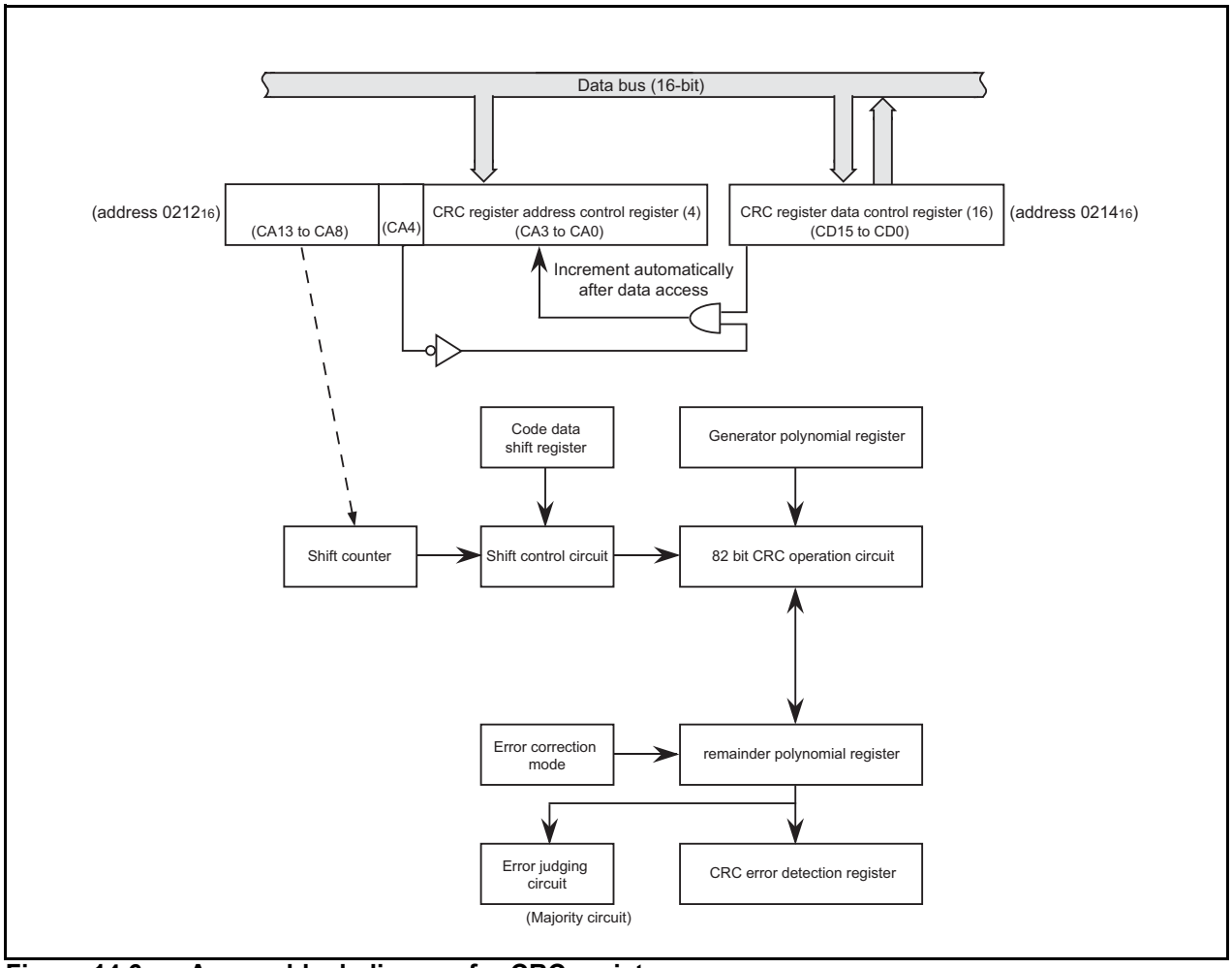



**Figure 14.6 Access block diagram for CRC registers**

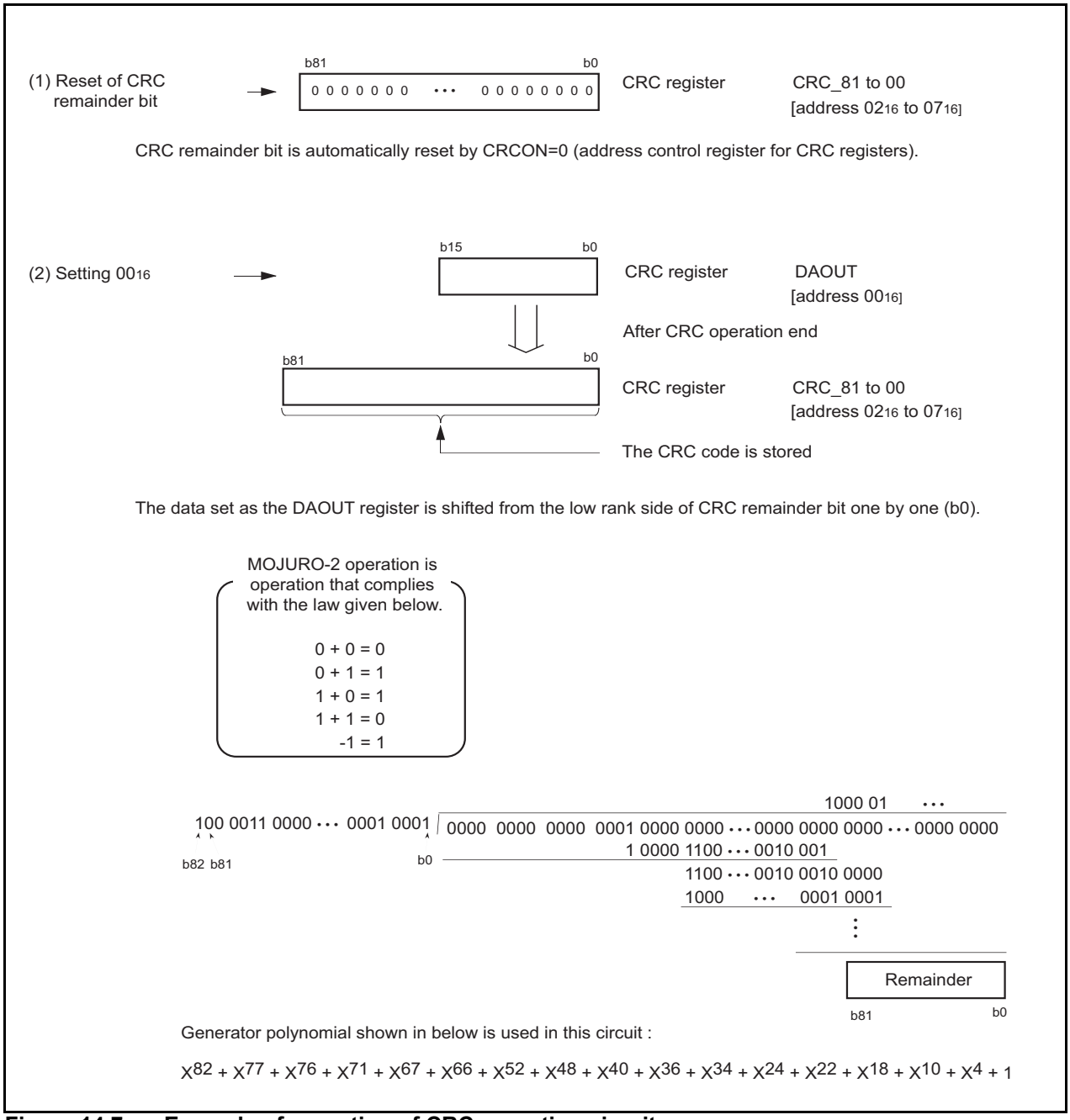



**Figure 14.7 Example of operation of CRC operation circuit**

```
;=============================================================================================================
;               Equations (Constant definition)
;=============================================================================================================
_CRC_ADRS                   .equ            00212h                  ; SFR address of CRC register address control register
_CRC_DATA                   .equ            00214h                  ; SFR address of CRC register data control register
SLICE_WORD_NUM              .equ            17                      ; Code data length (in nuits of word)

;=============================================================================================================
;               Macro definition
;=============================================================================================================
_wait       .macro
            nop
            nop
            nop
.endm

;=============================================================================================================
;               CRC operation routine
;=============================================================================================================

;--------- Setting of generator polynomial ---------------------------------------------------------------------------------------------------------------------------------------------
            mov.w       #0008H                  , _CRC_ADRS             ; Set the head address of the generator polynomial register
            _wait                                                       ; Wait
            mov.w       0000110000100011B       , _CRC_DATA             ; Coefficient of generator polynomial 82nd to 66th ( x^77 +x~76 +x^71 +x^67 +x^66)

;------ Writing of code data ---------------------------------------------------------------------------------------------------------------------------------------------------------
            mov.w       #0000H                  , _CRC_ADRS             ; Initialization of CRC register address control register
            mov.w       #9010H                  , _CRC_ADRS             ; Set up of CRCON=1, CRCCHANGE=0, CRCLOOP=10H, Increment=ON, and CRC address=00H.
            mov.w       #0000H                  , A0                    ; Initialization of a loop variable (A0)

L18:                                                                    ; Branch label
            cmp.w       #SLICE_WORD_NUM*2       , A0                    ; Comparison of the loop variable
            jgeu        L20                                             ; Go to L20 if writing code data is finished.
            lde.w       _CrcCodeData[A0]        , _CRC_DATA             ; Writing code data to the code data shift register.
            add.w       #0002H                  ,A0                     ; Increment of the address storing code data.
            jmp         L18                                             ; Return to the head of this loop.
L20:                                                                    ; Branch label
;--------- Dummy shift ----------------------------------------------------------------------------------------------------------------------------------------------------------------
            ; After finishing writing 272-bit code data,
            ; shift a bit for dummy surely in error correction mode.
            ; Specifying 1-bit  is set up by CRCLOOP=01H.
            mov.w       #8100H                  , _CRC_ADRS             ; Set up of CRCON=1, CRCCHANGE=0, CRCLOOP=10H, Increment=OFF, and CRC address=00H.
            _wait                                                       ; Wait
            mov.w       #0000H                  , _CRC_DATA             ; Writing data to the code data shift register for dummy shift.

;--------- Error detection ------------------------------------------------------------------------------------------------------------------------------------------------------------
; Since the address automatic increment in dummy shift (Increment=OFF), set CRC address=01H here.
; When accessing other CRC registers, the processing shown in the following two lines is necessary.
;           mov.w       #9001H      , _CRC_ADRS                         ; Set up of CRCON=1, CRCCHANGE=0, CRCLOOP=10H, Incremet=OFF and CRC address=01H.
;           _wait                                                       ; Wait

            mov.w       _CRC_DATA               , R0                    ; Read of CRC error detection register.
            cmp.w       #0000H                  , R0                    ; Judgement of CRC error.
            jeq                 L16                                     ; In the case of R0=0, branch to L16 since CRC error has not occurred (CRC error correction is skipped).

;--------- Error correction -----------------------------------------------------------------------------------------------------------------------------------------------------------
            mov.w       #0D010H                 , _CRC_ADRS             ; Set up of CRCON=1, CRCCHANGE=1, CRCLOOP=10H, Increment=ON and CRC address=00H.
            _wait                                                       ; Wait
            mov.w       #0000H                  , A0                    ; Initialization of a loop variable (A0)
L22:                                                                    ; Branch label
            cmp.w       #SLICE_WORD_NUB         , A0                    ; Comparison of the loop variable
            jgeu        L24                                             ; Go to L24 if correction of error data is finished.
            lde.w       _CrcCodeData[A0]        , _CRC_DATA             ; Writing code data to the code data shift register.
            jsr         _waitlong                                       ; Wait for finish of error correction.
            mov.w       _CRC_DATA               , _CrcCodeData[A0]      ; Read of error correction data in the address storing code data.
            add.w       #0002H                  , A0                    ; Increment of the address storing code data.
            jmp         L22                                             ; Return to the head of this loop.
L24:                                                                    ; Branch label

;------- The check of error correction data-----------------------------------------------------------------------------------------------------------------------------------------
            mov.w       #8111H                  , _CRC_ADRS             ; Set up of CRCON=1, CRCCHANGE=0, CRCLOOP=10H, Increment=ON and CRC address=00H
            _wait                                                       ; Wait
            mov.w       _CRC_DATA               , R0                    ; Error check after error correction. R0=000H if correction is performed.
L16:

;=============================================================================================================
; The function sample for weight for error correction
;=============================================================================================================
            .align
            .glb        _waitlong
_waitlong:                                                              ; Function label
            rts
```

**Figure 14.8 Example of program**

1.Bit composition of a CRC register

1. Address $00_{16}$ (=CA3 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DAOUT0 | The code data shift register write-in bit 0 | When write, data is written to "code data shift register" (Note).<br><br>When read, data differs between in error detection mode and in error correction mode.<br><br>• In error detection mode (CRCCHANGE=0)<br>$0000_{16}$ is read after shift end.<br>When read during shift operation, its content is indeterminate.<br><br>• In error correction mode (CRCCHANGE=1)<br>Corrected data is read after the original data is written in and some interval of data shift. | ○ | ○ |
| DAOUT1 | The code data shift register write-in bit 1 | | ○ | ○ |
| DAOUT2 | The code data shift register write-in bit 2 | | ○ | ○ |
| DAOUT3 | The code data shift register write-in bit 3 | | ○ | ○ |
| DAOUT4 | The code data shift register write-in bit 4 | | ○ | ○ |
| DAOUT5 | The code data shift register write-in bit 5 | | ○ | ○ |
| DAOUT6 | The code data shift register write-in bit 6 | | ○ | ○ |
| DAOUT7 | The code data shift register write-in bit 7 | | ○ | ○ |
| DAOUT8 | The code data shift register write-in bit 8 | | ○ | ○ |
| DAOUT9 | The code data shift register write-in bit 9 | | ○ | ○ |
| DAOUT10 | The code data shift register write-in bit 10 | | ○ | ○ |
| DAOUT11 | The code data shift register write-in bit 11 | | ○ | ○ |
| DAOUT12 | The code data shift register write-in bit 12 | | ○ | ○ |
| DAOUT13 | The code data shift register write-in bit 13 | | ○ | ○ |
| DAOUT14 | The code data shift register write-in bit 14 | | ○ | ○ |
| DAOUT15 | The code data shift register write-in bit 15 | | ○ | ○ |

Note: Refer to Figure 14.16 Access block diagram for CRC registers.

2. Address $01_{16}$ (=CA3 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CRC_ERR00 | The CRC bit 81 to 74 error detection bit | Logical OR of the CRC remainder bits 81 to 74 (address $02_{16}$) | ○ | × |
| CRC_ERR01 | The CRC bit 73 to 66 error detection bit | Logical OR of the CRC remainder bits 73 to 66 (address $02_{16}$) | ○ | × |
| CRC_ ERR02 | The CRC bit 65 to 58 error detection bit | Logical OR of the CRC remainder bits 65 to 58 (address $03_{16}$) | ○ | × |
| CRC_ERR03 | The CRC bit 57 to 50 error detection bit | Logical OR of the CRC remainder bits 57 to 50 (address $03_{16}$) | ○ | × |
| CRC_ERR04 | The CRC bit 49 to 42 error detection bit | Logical OR of the CRC remainder bits 49 to 42 (address $04_{16}$) | ○ | × |
| CRC_ERR05 | The CRC bit 41 to 34 error detection bit | Logical OR of the CRC remainder bits 41 to 34 (address $04_{16}$) | ○ | × |
| CRC_ERR06 | The CRC bit 33 to 26 error detection bit | Logical OR of the CRC remainder bits 33 to 26 (address $05_{16}$) | ○ | × |
| CRC_ERR07 | The CRC bit 25 to 18 error detection bit | Logical OR of the CRC remainder bits 25 to 18 (address $05_{16}$) | ○ | × |
| CRC_ERR08 | The CRC bit 17 to 10 error detection bit | Logical OR of the CRC remainder bits 17 to 10 (address $06_{16}$) | ○ | × |
| CRC_ERR09 | The CRC bit 09 to 02 error detection bit | Logical OR of the CRC remainder bits 09 to 02 (address $06_{16}$) | ○ | × |
| CRC_ERR10 | The CRC bit 01 to 00 error detection bit | Logical OR of the CRC remainder bits 01 to 00 (address $07_{16}$) | ○ | × |
| | Nothing is assigned. The value is "0" when it reads. | | × | × |

3. Address $02_{16}$ (=CA3 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CRC_81 | 81th remainder polynomial coefficient bit | The coefficient of each degree of a remainder polynomial is set up. It is shown in below when a remainder polynomial is made into CRC_MOD. $CRC_MOD = \sum_{n=0}^{81} CRC_n \cdot X^n$ | ○ | × |
| CRC_80 | 80th remainder polynomial coefficient bit | | ○ | × |
| CRC_79 | 79th remainder polynomial coefficient bit | | ○ | × |
| CRC_78 | 78th remainder polynomial coefficient bit | | ○ | × |
| CRC_77 | 77th remainder polynomial coefficient bit | | ○ | × |
| CRC_76 | 76th remainder polynomial coefficient bit | | ○ | × |
| CRC_75 | 75th remainder polynomial coefficient bit | | ○ | × |
| CRC_74 | 74th remainder polynomial coefficient bit | | ○ | × |
| CRC_73 | 73th remainder polynomial coefficient bit | | ○ | × |
| CRC_72 | 72th remainder polynomial coefficient bit | | ○ | × |
| CRC_71 | 71th remainder polynomial coefficient bit | | ○ | × |
| CRC_70 | 70th remainder polynomial coefficient bit | | ○ | × |
| CRC_69 | 69th remainder polynomial coefficient bit | | ○ | × |
| CRC_68 | 68th remainder polynomial coefficient bit | | ○ | × |
| CRC_67 | 67th remainder polynomial coefficient bit | | ○ | × |
| CRC_66 | 66th remainder polynomial coefficient bit | | ○ | × |

4. Address $03_{16}$ (=CA3 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CRC_65 | 65th remainder polynomial coefficient bit | Refer to CRC_81 to 66 (address $02_{16}$). | ○ | × |
| CRC_64 | 64th remainder polynomial coefficient bit | | ○ | × |
| CRC_63 | 63th remainder polynomial coefficient bit | | ○ | × |
| CRC_62 | 62th remainder polynomial coefficient bit | | ○ | × |
| CRC_61 | 61th remainder polynomial coefficient bit | | ○ | × |
| CRC_60 | 60th remainder polynomial coefficient bit | | ○ | × |
| CRC_59 | 59th remainder polynomial coefficient bit | | ○ | × |
| CRC_58 | 58th remainder polynomial coefficient bit | | ○ | × |
| CRC_57 | 57th remainder polynomial coefficient bit | | ○ | × |
| CRC_56 | 56th remainder polynomial coefficient bit | | ○ | × |
| CRC_55 | 55th remainder polynomial coefficient bit | | ○ | × |
| CRC_54 | 54th remainder polynomial coefficient bit | | ○ | × |
| CRC_53 | 53th remainder polynomial coefficient bit | | ○ | × |
| CRC_52 | 52th remainder polynomial coefficient bit | | ○ | × |
| CRC_51 | 51th remainder polynomial coefficient bit | | ○ | × |
| CRC_50 | 50th remainder polynomial coefficient bit | | ○ | × |

5. Address $04_{16}$ (=CA3 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CRC_49 | 49th remainder polynomial coefficient bit | Refer to CRC_81 to 66 (address $02_{16}$). | ○ | × |
| CRC_48 | 48th remainder polynomial coefficient bit | | ○ | × |
| CRC_47 | 47th remainder polynomial coefficient bit | | ○ | × |
| CRC_46 | 46th remainder polynomial coefficient bit | | ○ | × |
| CRC_45 | 45th remainder polynomial coefficient bit | | ○ | × |
| CRC_44 | 44th remainder polynomial coefficient bit | | ○ | × |
| CRC_43 | 43th remainder polynomial coefficient bit | | ○ | × |
| CRC_42 | 42th remainder polynomial coefficient bit | | ○ | × |
| CRC_41 | 41th remainder polynomial coefficient bit | | ○ | × |
| CRC_40 | 40th remainder polynomial coefficient bit | | ○ | × |
| CRC_39 | 39th remainder polynomial coefficient bit | | ○ | × |
| CRC_38 | 38th remainder polynomial coefficient bit | | ○ | × |
| CRC_37 | 37th remainder polynomial coefficient bit | | ○ | × |
| CRC_36 | 36th remainder polynomial coefficient bit | | ○ | × |
| CRC_35 | 35th remainder polynomial coefficient bit | | ○ | × |
| CRC_34 | 34th remainder polynomial coefficient bit | | ○ | × |

6. Address $05_{16}$ (=CA3 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CRC_33 | 33th remainder polynomial coefficient bit | Refer to CRC_81 to 66 (address $02_{16}$). | ○ | × |
| CRC_32 | 32th remainder polynomial coefficient bit | | ○ | × |
| CRC_31 | 31th remainder polynomial coefficient bit | | ○ | × |
| CRC_30 | 30th remainder polynomial coefficient bit | | ○ | × |
| CRC_29 | 29th remainder polynomial coefficient bit | | ○ | × |
| CRC_28 | 28th remainder polynomial coefficient bit | | ○ | × |
| CRC_27 | 27th remainder polynomial coefficient bit | | ○ | × |
| CRC_26 | 26th remainder polynomial coefficient bit | | ○ | × |
| CRC_25 | 25th remainder polynomial coefficient bit | | ○ | × |
| CRC_24 | 24th remainder polynomial coefficient bit | | ○ | × |
| CRC_23 | 23th remainder polynomial coefficient bit | | ○ | × |
| CRC_22 | 22th remainder polynomial coefficient bit | | ○ | × |
| CRC_21 | 21th remainder polynomial coefficient bit | | ○ | × |
| CRC_20 | 20th remainder polynomial coefficient bit | | ○ | × |
| CRC_19 | 19th remainder polynomial coefficient bit | | ○ | × |
| CRC_18 | 18th remainder polynomial coefficient bit | | ○ | × |

7. Address $06_{16}$ (=CA3 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CRC_17 | 17th remainder polynomial coefficient bit | Refer to CRC_81 to 66 (address $02_{16}$). | ○ | × |
| CRC_16 | 16th remainder polynomial coefficient bit | | ○ | × |
| CRC_15 | 15th remainder polynomial coefficient bit | | ○ | × |
| CRC_14 | 14th remainder polynomial coefficient bit | | ○ | × |
| CRC_13 | 13th remainder polynomial coefficient bit | | ○ | × |
| CRC_12 | 12th remainder polynomial coefficient bit | | ○ | × |
| CRC_11 | 11th remainder polynomial coefficient bit | | ○ | × |
| CRC_10 | 10th remainder polynomial coefficient bit | | ○ | × |
| CRC_09 | 09th remainder polynomial coefficient bit | | ○ | × |
| CRC_08 | 08th remainder polynomial coefficient bit | | ○ | × |
| CRC_07 | 07th remainder polynomial coefficient bit | | ○ | × |
| CRC_06 | 06th remainder polynomial coefficient bit | | ○ | × |
| CRC_05 | 05th remainder polynomial coefficient bit | | ○ | × |
| CRC_04 | 04th remainder polynomial coefficient bit | | ○ | × |
| CRC_03 | 03rd remainder polynomial coefficient bit | | ○ | × |
| CRC_02 | 02nd remainder polynomial coefficient bit | | ○ | × |

8. Address $07_{16}$ (=CA3 to 0)



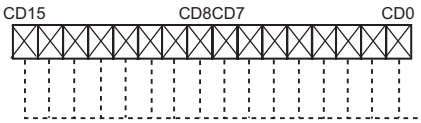

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CRC_01 | 01st remainder polynomial coefficient bit | Refer to CRC_81 to 66 (address $02_{16}$). | ○ | × |
| CRC_00 | 00th remainder polynomial coefficient bit | | ○ | × |
| | Nothing is assigned. The value is "0" when it reads. | | × | × |

9. Address $08_{16}$ (=CA3 to 0)



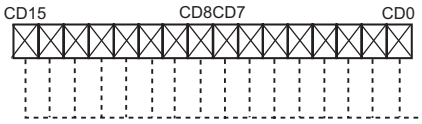

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| REG_C0 | Coefficient bit of the 66/0th generation polynomials | The coefficient of each degree of the generation polynomial is set.<br>The coefficient from 81 to the 66th is set for 82 bit CRC mode.<br>The 15th in case of 16 bit CRC mode. The 0th coefficients are set.<br>When generation polynomial to be CRC_GP,<br>$CRC_GP = X^{82} + \sum_{n=0}^{15} REG_Cn \cdot X^{n+66} + X^{56} + X^{52} + X^{48} + X^{40} + X^{36} + X^{34} + X^{24} + X^{22} + X^{18} + X^{10} + X^{4} + 1$<br>(For 82 bit CRC mode)<br>$CRC_GP = X^{16} + \sum_{n=0}^{15} RE6_Cn \cdot X^{n}$<br>(For 16 bit CRC mode) | ○ | ○ |
| REG_C1 | Coefficient bit of the 67/1th generation polynomials | | ○ | ○ |
| REG_C2 | Coefficient bit of the 68/2th generation polynomials | | ○ | ○ |
| REG_C3 | Coefficient bit of the 69/3th generation polynomials | | ○ | ○ |
| REG_C4 | Coefficient bit of the 70/4th generation polynomials | | ○ | ○ |
| REG_C5 | Coefficient bit of the 71/5th generation polynomials | | ○ | ○ |
| REG_C6 | Coefficient bit of the 72/6th generation polynomials | | ○ | ○ |
| REG_C7 | Coefficient bit of the 73/7th generation polynomials | | ○ | ○ |
| REG_C8 | Coefficient bit of the 74/8th generation polynomials | | ○ | ○ |
| REG_C9 | Coefficient bit of the 75/9th generation polynomials | | ○ | ○ |
| REG_C10 | Coefficient bit of the 76/10th generation polynomials | | ○ | ○ |
| REG_C11 | Coefficient bit of the 77/11th generation polynomials | | ○ | ○ |
| REG_C12 | Coefficient bit of the 78/12th generation polynomials | | ○ | ○ |
| REG_C13 | Coefficient bit of the 79/13th generation polynomials | | ○ | ○ |
| REG_C14 | Coefficient bit of the 80/14th generation polynomials | | ○ | ○ |
| REG_C15 | Coefficient bit of the 81/15th generation polynomials | | ○ | ○ |

10. Address $09_{16}$ (=CA3 to 0)



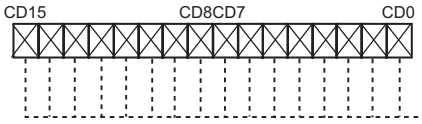

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| | Nothing is assigned.<br>The value is unfixed when it reads. | | × | × |

11. Address $0A_{16}$ (=CA3 to 0)



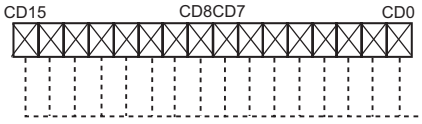

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| | | Nothing is assigned.<br>The value is unfixed when it reads. | × | × |

12. Address $0B_{16}$ (=CA3 to 0)



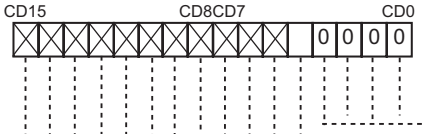

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| | | Nothing is assigned.<br>The value is unfixed when it reads. | × | × |

13. Address $0C_{16}$ (=CA3 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| | | Nothing is assigned.<br>The value is unfixed when it reads. | × | × |

14. Address $0D_{16}$ (=CA3 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Reserved bit | | Must set to "0." | ○ | ○ |
| CRC16SEL | CRC mode select bit | Selects 82 bit CRC/16 bit CRC mode<br>CRC16SEL: 0 = 82 bit CRC; 1 = 16 bit CRC | ○ | ○ |
| | | Nothing is assigned Indeterminate at reading | × | × |

## 14.5 Expansion Register

Control Data slice function. Expansion register composition is shown in Table 14.4.

**Table 14.4 Expansion register composition**

| DA5 to DA0 | DD15 | DD14 | DD13 | DD12 | DD11 | DD10 | DD9 | DD8 | DD7 | DD6 | DD5 | DD4 | DD3 | DD2 | DD1 | DD0 | Remarks |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 00H | LN15_EV0 | LN14_EV0 | LN13_EV0 | LN12_EV0 | LN11_EV0 | LN10_EV0 | LN9_EV0 | LN8_EV0 | LN7_EV0 | LN6_EV0 | LN5_EV0 | LN4_EV0 | LN3_EV0 | LN2_EV0 | LN1_EV0 | LN0_EV0 | Line register |
| 01H | LN15_EV1 | LN14_EV1 | LN13_EV1 | LN12_EV1 | LN11_EV1 | LN10_EV1 | LN9_EV1 | LN8_EV1 | LN7_EV1 | LN6_EV1 | LN5_EV1 | LN4_EV1 | LN3_EV1 | LN2_EV1 | LN1_EV1 | LN0_EV1 | |
| 02H | LN17_EV0 | LN16_EV0 | – | – | – | – | – | – | LN17_OD0 | LN16_OD0 | – | – | – | – | – | – | |
| 03H | LN17_EV1 | LN16_EV1 | – | – | – | – | – | – | LN17_OD1 | LN16_OD1 | – | – | – | – | – | – | |
| 04H | LN15_OD0 | LN14_OD0 | LN13_OD0 | LN12_OD0 | LN11_OD0 | LN10_OD0 | LN9_OD0 | LN8_OD0 | LN7_OD0 | LN6_OD0 | LN5_OD0 | LN4_OD0 | LN3_OD0 | LN2_OD0 | LN1_OD0 | LN0_OD0 | Status register 1 |
| 05H | LN15_OD1 | LN14_OD1 | LN13_OD1 | LN12_OD1 | LN11_OD1 | LN10_OD1 | LN9_OD1 | LN8_OD1 | LN7_OD1 | LN6_OD1 | LN5_OD1 | LN4_OD1 | LN3_OD1 | LN2_OD1 | LN1_OD1 | LN0_OD1 | |
| 06H | DIVS1 | DIVS0 | SELVCO | – | – | – | – | – | – | – | – | – | – | – | – | – | |
| 07H | FLC15 | FLC14 | FLC13 | FLC12 | FLC11 | FLC10 | FLC9 | FLC8 | FLC7 | FLC6 | FLC5 | FLC4 | FLC3 | FLC2 | FLC1 | FLC0 | |
| 08H | CHK_FLC15 | CHK_FLC14 | CHK_FLC13 | CHK_FLC12 | CHK_FLC11 | CHK_FLC10 | CHK_FLC9 | CHK_FLC8 | CHK_FLC7 | CHK_FLC6 | CHK_FLC5 | CHK_FLC4 | CHK_FLC3 | CHK_FLC2 | CHK_FLC1 | CHK_FLC0 | |
| 09H | – | – | – | – | BIFON | SLSLVL1 | SLSLVL0 | – | SEKI7 | SEKI6 | SEKI5 | SEKI4 | SEKI3 | SEKI2 | SEKI1 | SEKI0 | |
| 0AH | GETPEEK3 | GETPEEK2 | GETPEEK1 | GETPEEK0 | – | – | GET_HP1 | GET_HP0 | SLS_HP7 | SLS_HP6 | SLS_HP5 | SLS_HP4 | SLS_HP3 | SLS_HP2 | SLS_HP1 | SLS_HP0 | |
| 0BH | – | FRAM | – | – | – | – | – | – | – | – | – | – | – | – | – | – | |
| 0CH | – | GSTTIM | SELSTART | – | – | – | – | – | SLS7 | SLS6 | SLS5 | SLS4 | SLS3 | SLS2 | SLS1 | SLS0 | |
| 0DH | DIVS1 | DIVS0 | SELVCO | – | – | – | – | – | – | – | – | – | – | – | – | – | |
| 0EH | FLC15 | FLC14 | FLC13 | FLC12 | FLC11 | FLC10 | FLC9 | FLC8 | FLC7 | FLC6 | FLC5 | FLC4 | FLC3 | FLC2 | FLC1 | FLC0 | |
| 0FH | CHK_FLC15 | CHK_FLC14 | CHK_FLC13 | CHK_FLC12 | CHK_FLC11 | CHK_FLC10 | CHK_FLC9 | CHK_FLC8 | CHK_FLC7 | CHK_FLC6 | CHK_FLC5 | CHK_FLC4 | CHK_FLC3 | CHK_FLC2 | CHK_FLC1 | CHK_FLC0 | Status register 2 |
| 10H | – | – | – | – | BIFON | SLSLVL1 | SLSLVL0 | – | SEKI7 | SEKI6 | SEKI5 | SEKI4 | SEKI3 | SEKI2 | SEKI1 | SEKI0 | |
| 11H | GETPEEK3 | GETPEEK2 | GETPEEK1 | GETPEEK0 | – | – | GET_HP1 | GET_HP0 | SLS_HP7 | SLS_HP6 | SLS_HP5 | SLS_HP4 | SLS_HP3 | SLS_HP2 | SLS_HP1 | SLS_HP0 | |
| 12H | – | FRAM | – | – | – | – | – | – | – | – | – | – | – | – | – | – | |
| 13H | – | GSTTIM | SELSTART | – | – | – | – | – | SLS7 | SLS6 | SLS5 | SLS4 | SLS3 | SLS2 | SLS1 | SLS0 | |
| 14H | DIVS1 | DIVS0 | SELVCO | – | – | – | – | – | – | – | – | – | – | – | – | – | |
| 15H | FLC15 | FLC14 | FLC13 | FLC12 | FLC11 | FLC10 | FLC9 | FLC8 | FLC7 | FLC6 | FLC5 | FLC4 | FLC3 | FLC2 | FLC1 | FLC0 | |
| 16H | CHK_FLC15 | CHK_FLC14 | CHK_FLC13 | CHK_FLC12 | CHK_FLC11 | CHK_FLC10 | CHK_FLC9 | CHK_FLC8 | CHK_FLC7 | CHK_FLC6 | CHK_FLC5 | CHK_FLC4 | CHK_FLC3 | CHK_FLC2 | CHK_FLC1 | CHK_FLC0 | Status register 3 |
| 17H | – | – | – | – | BIFON | SLSLVL1 | SLSLVL0 | – | SEKI7 | SEKI6 | SEKI5 | SEKI4 | SEKI3 | SEKI2 | SEKI1 | SEKI0 | |
| 18H | GETPEEK3 | GETPEEK2 | GETPEEK1 | GETPEEK0 | – | – | GET_HP1 | GET_HP0 | SLS_HP7 | SLS_HP6 | SLS_HP5 | SLS_HP4 | SLS_HP3 | SLS_HP2 | SLS_HP1 | SLS_HP0 | |
| 19H | – | FRAM | – | – | – | – | – | – | – | – | – | – | – | – | – | – | |
| 1AH | – | GSTTIM | SELSTART | – | – | – | – | – | SLS7 | SLS6 | SLS5 | SLS4 | SLS3 | SLS2 | SLS1 | SLS0 | |
| 1BH | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | |
| 1CH | ADSTART | – | EXT_PDC2 | – | – | – | – | – | – | – | – | – | – | – | – | – | |
| 1DH | – | – | – | – | – | – | VPS_VCO_ON | PDC_VCO_R1 | PDC_VCO_R0 | PDC_VCO_ON | – | – | – | – | – | – | |
| 1EH | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | for read |
| 1FH | – | – | – | – | – | – | – | – | MACRO_ON | – | – | FLD1V | – | – | – | – | for read |
| 20H | – | MPAL | NXP | – | – | – | LEVELA | NORMAL | – | SEPV0 | – | – | – | – | – | – | |
| 21H | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | |
| 22H | HM84SEL | – | – | – | DIV_PDC8 | DIV_PDC7 | DIV_PDC6 | DIV_PDC5 | DIV_PDC4 | DIV_PDC3 | DIV_PDC2 | DIV_PDC1 | DIV_PDC0 | DIV_PDCS2 | DIV_PDCS1 | DIV_PDCS0 | |
| 23H | – | – | – | HORAX_ON | DIV_VPS8 | DIV_VPS7 | DIV_VPS6 | DIV_VPS5 | DIV_VPS4 | DIV_VPS3 | DIV_VPS2 | DIV_VPS1 | DIV_VPS0 | DIV_VPSS2 | DIV_VPSS1 | DIV_VPSS0 | |
| 24H | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | |
| 25H | – | – | – | – | – | – | – | – | – | – | – | SLION_TIM | – | – | ADON_TIM | ADSEL | |
| 26H | DIVV_CK7 | DIVV_CK6 | DIVV_CK5 | DIVV_CK4 | DIVV_CK3 | DIVV_CK2 | DIVV_CK1 | DIVV_CK0 | DIVP_CK7 | DIVP_CK6 | DIVP_CK5 | DIVP_CK4 | DIVP_CK3 | DIVP_CK2 | DIVP_CK1 | DIVP_CK0 | |
| 27H | – | – | – | WEIGHT4 | WEIGHT3 | WEIGHT2 | WEIGHT1 | WEIGHT0 | DLYSEL7 | DLYSEL6 | DLYSEL5 | DLYSEL4 | DLYSEL3 | DLYSEL2 | DLYSEL1 | DLYSEL0 | |
| 28H | – | – | – | – | – | – | – | INTAD | ADON | SYNLVL2 | SYNLVL1 | SYNLVL0 | – | 6BITOFF | START | ADLAT | |
| 29H | – | – | – | – | STBSYNCSEP | SYNCSEP_ONG | SLI_GO | VPS_VP8 | VPS_VP7 | VPS_VP6 | VPS_VP5 | VPS_VP4 | VPS_VP3 | VPS_VP2 | VPS_VP1 | VPS_VP0 | |
| 2AH | – | – | – | – | – | – | – | – | MASK7 | MASK6 | MASK5 | MASK4 | MASK3 | MASK2 | MASK1 | MASK0 | |
| 2BH | SEL_PDEC | – | SEL_VPSH | SEL_PDCH | – | – | – | – | – | – | – | – | – | – | – | – | |
| 2CH | – | – | – | – | – | – | – | PLSPOS8 | PLSPOS7 | PLSPOS6 | PLSPOS5 | PLSPOS4 | PLSPOS3 | PLSPOS2 | PLSPOS1 | PLSPOS0 | |
| 2DH | – | – | – | – | – | – | – | PLSNEG8 | PLSNEG7 | PLSNEG6 | PLSNEG5 | PLSNEG4 | PLSNEG3 | PLSNEG2 | PLSNEG1 | PLSNEG0 | |
| 2EH | HCOUNT15 | HCOUNT14 | HCOUNT13 | HCOUNT12 | HCOUNT11 | HCOUNT10 | HCOUNT9 | HCOUNT8 | HCOUNT7 | HCOUNT6 | HCOUNT5 | HCOUNT4 | HCOUNT3 | HCOUNT2 | HCOUNT1 | HCOUNT0 | for read |
| 2FH | STB_RES | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | |
| 30H | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | |
| 31H | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | |
| 32H | RMTSEL | – | – | – | JSTCKON | JSTCKDIV1 | JSTCKDIV0 | RMTHD8(8) | RMTHD8(7) | RMTHD8(6) | RMTHD8(5) | RMTHD8(4) | RMTHD8(3) | RMTHD8(2) | RMTHD8(1) | RMTHD8(0) | |
| 33H | FILDIV1(1) | FILDIV1(0) | – | VERTX | FILDIV0 | – | FILDIV0 | RMTHD1(8) | RMTHD1(7) | RMTHD1(6) | RMTHD1(5) | RMTHD1(4) | RMTHD1(3) | RMTHD1(2) | RMTHD1(1) | RMTHD1(0) | |
| 34H | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | |
| 35H | EXAOFF | – | – | – | – | PTD8 | PTC8 | HINT_LINE8 | HINT_LINE7 | HINT_LINE6 | HINT_LINE5 | HINT_LINE4 | HINT_LINE3 | HINT_LINE2 | HINT_LINE1 | HINT_LINE0 | |
| 36H | SECINT3 | SECINT2 | SECINT1 | SECINT0 | HINT3 | HINT2 | HINT1 | HINT0 | INTRMT3 | INTRMT2 | INTRMT1 | INTRMT0 | VINT3 | VINT2 | VINT1 | VINT0 | |
| 37H | – | – | YUKOU1(5) | YUKOU1(4) | YUKOU1(3) | YUKOU1(2) | YUKOU1(1) | YUKOU1(0) | – | – | YUKOU0(5) | YUKOU0(4) | YUKOU0(3) | YUKOU0(2) | YUKOU0(1) | YUKOU0(0) | |
| 38H | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | |
| 39H | SECJUST | – | – | – | – | – | – | RTCON | – | – | SECOUT5 | SECOUT4 | SECOUT3 | SECOUT2 | SECOUT1 | SECOUT0 | |
| 3AH | – | – | – | – | – | MINOUT10 | MINOUT9 | MINOUT8 | MINOUT7 | MINOUT6 | MINOUT5 | MINOUT4 | MINOUT3 | MINOUT2 | MINOUT1 | MINOUT0 | |
| 3BH | DAYCUONT15 | DAYCUONT14 | DAYCUONT13 | DAYCUONT12 | DAYCUONT11 | DAYCUONT10 | DAYCUONT9 | DAYCUONT8 | DAYCUONT7 | DAYCUONT6 | DAYCUONT5 | DAYCUONT4 | DAYCUONT3 | DAYCUONT2 | DAYCUONT1 | DAYCUONT0 | |
| 3CH | RMT_TM(7) | RMT_TM(6) | RMT_TM(5) | RMT_TM(4) | RMT_TM(3) | RMT_TM(2) | RMT_TM(1) | RMT_TM(0) | RMTDIV1(1) | RMTDIV0(0) | RMCDIV(1) | RMCDIV(0) | IRPOL(2) | IRPOL(1) | IRPOL(0) | RMTSTART | |
| 3DH | RMT_TMH(7) | RMT_TMH(6) | RMT_TMH(5) | RMT_TMH(4) | RMT_TMH(3) | RMT_TMH(2) | RMT_TMH(1) | RMT_TMH(0) | RMT_TML(7) | RMT_TML(6) | RMT_TML(5) | RMT_TML(4) | RMT_TML(3) | RMT_TML(2) | RMT_TML(1) | RMT_TML(0) | |
| 3EH | – | IROUT_SLICEON | – | – | – | – | – | – | – | – | – | – | RMTTXINT(3) | RMTTXINT(2) | RMTTXINT(1) | RMTTXINT(0) | |
| 3FH | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | – | |

For accessing to expansion register data, set accessing address (DA5 to DA0) (shown in Table 14.4) to expansion register address control register (address 0216$_{16}$). Then write data (DD15 to DD0) to expansion register data control register (address 0218$_{16}$). When end the data accessing, expansion register address control register increments address automatically. Then, next address data writing is possible. After reset, the value of expansion register become "0" all, except for the clock timer.
Expansion register access registers are shown in Figure 14.9, expansion register access block diagram is shown in Figure 14.10, and expansion register bit compositions are shown in p202 to p239.

Expansion register address control register

b15 b8 b7 b6 b0

| Symbol | Address | When reset |
|---|---|---|
| DA | 0216$_{16}$ | 0000$_{16}$ |

| Function | Setting possible value | R | W |
|---|---|---|---|
| Specify accessing expansion register address | 00$_{16}$ to 45$_{16}$ | ○ | ○ |
| Nothing is assigned.<br>When write, set to "0".<br>When read, its content is indeterminate. | | – | – |
| Expansion register address automatic increment<br>0:enable / 1:disable (Note2) | – | × | ○ |
| Nothing is assigned.<br>In an attempt to write to these bits, write "0". The value, if read, turns out to be indeterminated. | | – | – |

Note1 : When access to expansion register, must be set expansion register address at first, then use expansion register data control register (0218$_{16}$).
Note2 : When bit 8 = "0" setting, expansion register data control register increments by accessing expansion register data control register, so it is not necessary to setting the next expansion register address. When bit 8 = "1" setting, the address is fixed.

Expansion register data control register

b15 b8 b7 b0

| Symbol | Address | When reset |
|---|---|---|
| DD | 0218$_{16}$ | 0000$_{16}$ |

| Function | Setting possible value | R | W |
|---|---|---|---|
| Write and read out the data of expansion register which is specified by expansion register address control register (address 0216$_{16}$) | 0000$_{16}$ to FFFF$_{16}$ | ○ | ○ |

Note : Data access must be 16-bit unit. 8-bit unit access is disable.

**Figure 14.9 Expansion register access registers composition**

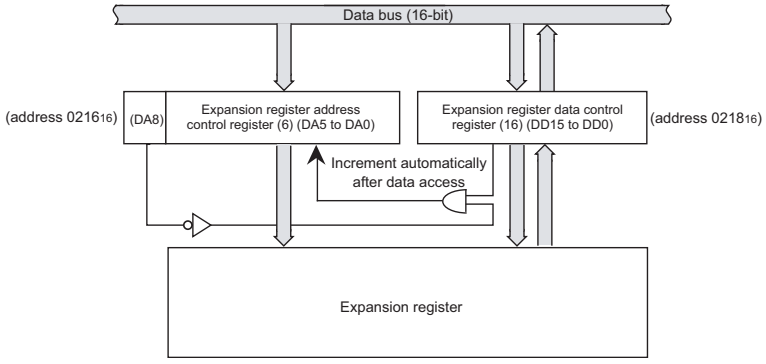



**Figure 14.10 Expansion register access block diagram**

Bit composition of an expansion register

1. Address $00_{16}$ (=DA5 to 0)



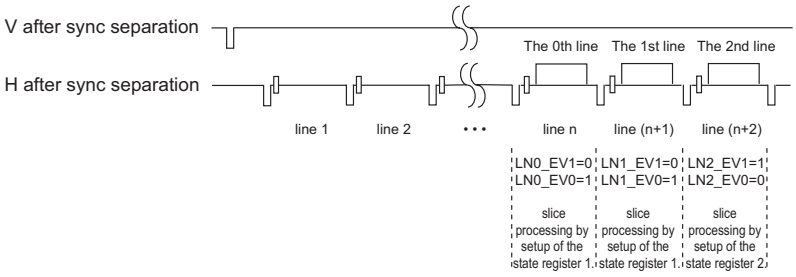

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| LN0_EV0 | The 0th line state register selection bit | As for the slicing method of the n-th line (Notes 1), it is chosen which set of the state register settings of the three sets (Notes 2) is used with the combination of LNn_EV0 (address $00_{16}$ and $02_{16}$, n = 0 to 17) and LNn_EV1 (address $01_{16}$ and $03_{16}$, n= 0 to 17.)<br><br>Four kinds of following state registers can be chosen for every line (Notes 3.) | ○ | ○ |
| LN1_EV0 | The 1st line state register selection bit | | ○ | ○ |
| LN2_EV0 | The 2nd line state register selection bit | | ○ | ○ |
| LN3_EV0 | The 3rd line state register selection bit | | ○ | ○ |
| LN4_EV0 | The 4th line state register selection bit | | ○ | ○ |
| LN5_EV0 | The 5th line state register selection bit | | ○ | ○ |
| LN6_EV0 | The 6th line state register selection bit | | ○ | ○ |
| LN7_EV0 | The 7th line state register selection bit | | ○ | ○ |
| LN8_EV0 | The 8th line state register selection bit | | ○ | ○ |
| LN9_EV0 | The 9th line state register selection bit | | ○ | ○ |
| LN10_EV0 | The 10th line state register selection bit | | ○ | ○ |
| LN11_EV0 | The 11th line state register selection bit | | ○ | ○ |
| LN12_EV0 | The 12th line state register selection bit | | ○ | ○ |
| LN13_EV0 | The 13th line state register selection bit | | ○ | ○ |
| LN14_EV0 | The 14th line state register selection bit | | ○ | ○ |
| LN15_EV0 | The 15th line state register selection bit | | ○ | ○ |

| LNn_EV1 | LNn_EV0 | State register(Notes 2) |
|---|---|---|
| 0 | 0 | Do not set up |
| 0 | 1 | State register 1 |
| 1 | 0 | State register 2 |
| 1 | 1 | State register 3 |

Notes 1. The n-th line: The number of lines after a slice start.
Please refer to the supplement (3) of 15.6 expansion register composition (P229) for details.

Notes 2. 06h to 0Ch address: State register 1
0Dh to 13h address: State register 2
14h to 1Ah address: State register 3

Notes 3. The example of a setting.

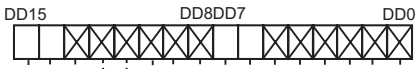


2. Address $01_{16}$ (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| LN0_EV1 | The 0th line state register selection bit | Refer to LNn_EV0 (address $00_{16}$) | ○ | ○ |
| LN1_EV1 | The 1st line state register selection bit | | ○ | ○ |
| LN2_EV1 | The 2nd line state register selection bit | | ○ | ○ |
| LN3_EV1 | The 3rd line state register selection bit | | ○ | ○ |
| LN4_EV1 | The 4th line state register selection bit | | ○ | ○ |
| LN5_EV1 | The 5th line state register selection bit | | ○ | ○ |
| LN6_EV1 | The 6th line state register selection bit | | ○ | ○ |
| LN7_EV1 | The 7th line state register selection bit | | ○ | ○ |
| LN8_EV1 | The 8th line state register selection bit | | ○ | ○ |
| LN9_EV1 | The 9th line state register selection bit | | ○ | ○ |
| LN10_EV1 | The 10th line state register selection bit | | ○ | ○ |
| LN11_EV1 | The 11th line state register selection bit | | ○ | ○ |
| LN12_EV1 | The 12th line state register selection bit | | ○ | ○ |
| LN13_EV1 | The 13th line state register selection bit | | ○ | ○ |
| LN14_EV1 | The 14th line state register selection bit | | ○ | ○ |
| LN15_EV1 | The 15th line state register selection bit | | ○ | ○ |

3. Address $02_{16}$ (=DA5 to 0)



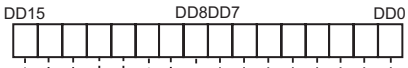

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Nothing is assigned. | | | × | × |
| LN16_OD0 | The 16th line state register selection bit | Refer to LNn_OD0 (address $04_{16}$) | ○ | ○ |
| LN17_OD0 | The 17th line state register selection bit | | ○ | ○ |
| Nothing is assigned. | | | × | × |
| LN16_EV0 | The 16th line state register selection bit | Refer to LNn_EV0 (address $00_{16}$) | ○ | ○ |
| LN17_EV0 | The 17th line state register selection bit | | ○ | ○ |

4. Address $03_{16}$ (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Nothing is assigned. | | | × | × |
| LN16_OD1 | The 16th line state register selection bit | Refer to LNn_OD0 (address $04_{16}$) | ○ | ○ |
| LN17_OD1 | The 17th line state register selection bit | | ○ | ○ |
| Nothing is assigned. | | | × | × |
| LN16_EV1 | The 16th line state register selection bit | Refer to LNn_EV0 (address $00_{16}$) | ○ | ○ |
| LN17_EV1 | The 17th line state register selection bit | | ○ | ○ |

5. Address $04_{16}$ (=DA5 to 0)



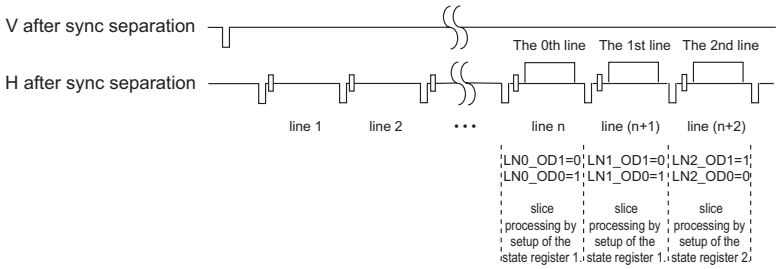

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| LN0_OD0 | The 0th line state register selection bit | As for the slicing method of the n-th line (Notes 1), it is chosen which set of the state register settings of the three sets (Notes 2) is used with the combination of LNn_OD0 (address $04_{16}$ and $02_{16}$, n = 0 to 17) and LNn_OD1 (address $05_{16}$ and $03_{16}$, n= 0 to 17.)<br><br>Four kinds of following state registers can be chosen for every line. (Notes 3) | ○ | ○ |
| LN1_OD0 | The 1st line state register selection bit | | ○ | ○ |
| LN2_OD0 | The 2nd line state register selection bit | | ○ | ○ |
| LN3_OD0 | The 3rd line state register selection bit | | ○ | ○ |
| LN4_OD0 | The 4th line state register selection bit | | ○ | ○ |
| LN5_OD0 | The 5th line state register selection bit | | ○ | ○ |
| LN6_OD0 | The 6th line state register selection bit | | ○ | ○ |
| LN7_OD0 | The 7th line state register selection bit | | ○ | ○ |
| LN8_OD0 | The 8th line state register selection bit | | ○ | ○ |
| LN9_OD0 | The 9th line state register selection bit | | ○ | ○ |
| LN10_OD0 | The 10th line state register selection bit | | ○ | ○ |
| LN11_OD0 | The 11th line state register selection bit | | ○ | ○ |
| LN12_OD0 | The 12th line state register selection bit | | ○ | ○ |
| LN13_OD0 | The 13th line state register selection bit | | ○ | ○ |
| LN14_OD0 | The 14th line state register selection bit | | ○ | ○ |
| LN15_OD0 | The 15th line state register selection bit | | ○ | ○ |

| LNn_EV1 | LNn_EV0 | State register(Notes 2) |
|---|---|---|
| 0 | 0 | Do not set up |
| 0 | 1 | State register 1 |
| 1 | 0 | State register 2 |
| 1 | 1 | State register 3 |

Notes 1. The n-th line: The number of lines after a slice start.
Please refer to the supplement (3) of 14.6 expansion register composition for details.

Notes 2. 06h to 0Ch address: State register 1
0Dh to 13h address: State register 2
14h to 1Ah address: State register 3

Notes 3. The example of a setting.

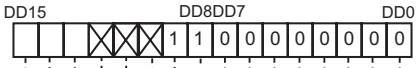


6. Address $05_{16}$ (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| LN0_OD1 | The 0th line state register selection bit | Refer to LNn_OD0 (address $04_{16}$) | ○ | ○ |
| LN1_OD1 | The 1st line state register selection bit | | ○ | ○ |
| LN2_OD1 | The 2nd line state register selection bit | | ○ | ○ |
| LN3_OD1 | The 3rd line state register selection bit | | ○ | ○ |
| LN4_OD1 | The 4th line state register selection bit | | ○ | ○ |
| LN5_OD1 | The 5th line state register selection bit | | ○ | ○ |
| LN6_OD1 | The 6th line state register selection bit | | ○ | ○ |
| LN7_OD1 | The 7th line state register selection bit | | ○ | ○ |
| LN8_OD1 | The 8th line state register selection bit | | ○ | ○ |
| LN9_OD1 | The 9th line state register selection bit | | ○ | ○ |
| LN10_OD1 | The 10th line state register selection bit | | ○ | ○ |
| LN11_OD1 | The 11th line state register selection bit | | ○ | ○ |
| LN12_OD1 | The 12th line state register selection bit | | ○ | ○ |
| LN13_OD1 | The 13th line state register selection bit | | ○ | ○ |
| LN14_OD1 | The 14th line state register selection bit | | ○ | ○ |
| LN15_OD1 | The 15th line state register selection bit | | ○ | ○ |

7. Address $06_{16}$, $0D_{16}$, $14_{16}$ (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Reserved bits | | Must set to "0." | × | ○ |
| Reserved bits | | Must set to "1." | × | ○ |
| Nothing is assigned. | | | × | × |
| SELVCO | The PLL selection bit for slice | 0: PDC<br>1: VPS | ○ | ○ |
| DIVS0 | The clock division bit for slice | DIVS1 DIVS0 divided value<br>0 0 no division<br>0 1 divided by 2<br>1 0 divided by 3<br>1 1 divided by 5 | ○ | ○ |
| DIVS1 | | | ○ | ○ |

8. Address $07_{16}$, $0E_{16}$, $15_{16}$ (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| FLC0 | Framing code selection bit | Framing code is set up<br>Clock run-in / Framing code / Data<br>Setup<br>FLC0 to FLC15<br>16 bits are checked at maximum. However, the bit of CHK_FLCn (addresses $08_{16}$, $0F_{16}$ and $16_{16}$) = "1" is not checked. | ○ | ○ |
| FLC1 | | | ○ | ○ |
| FLC2 | | | ○ | ○ |
| FLC3 | | | ○ | ○ |
| FLC4 | | | ○ | ○ |
| FLC5 | | | ○ | ○ |
| FLC6 | | | ○ | ○ |
| FLC7 | | | ○ | ○ |
| FLC8 | | | ○ | ○ |
| FLC9 | | | ○ | ○ |
| FLC10 | | | ○ | ○ |
| FLC11 | | | ○ | ○ |
| FLC12 | | | ○ | ○ |
| FLC13 | | | ○ | ○ |
| FLC14 | | | ○ | ○ |
| FLC15 | | | ○ | ○ |

9. Address 08$_{16}$, 0F$_{16}$, 16$_{16}$ (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CHK_FLC0 | Framing code check selection bit | When acquiring data, it sets up whether framing code set up by FLC 0 to 15 (addresses 07$_{16}$, 0E$_{16}$, and 15$_{16}$) is checked or not per bit.<br>Data will be acquired if the n-th bit which is set as check is in agreement. | ○ | ○ |
| CHK_FLC1 | | | ○ | ○ |
| CHK_FLC2 | | | ○ | ○ |
| CHK_FLC3 | | | ○ | ○ |
| CHK_FLC4 | | | ○ | ○ |
| CHK_FLC5 | | | ○ | ○ |
| CHK_FLC6 | | | ○ | ○ |
| CHK_FLC7 | | | ○ | ○ |
| CHK_FLC8 | | | ○ | ○ |
| CHK_FLC9 | | | ○ | ○ |
| CHK_FLC10 | | | ○ | ○ |
| CHK_FLC11 | | | ○ | ○ |
| CHK_FLC12 | | | ○ | ○ |
| CHK_FLC13 | | | ○ | ○ |
| CHK_FLC14 | | | ○ | ○ |
| CHK_FLC15 | | | ○ | ○ |

| CHK_FLCn | n-th bit |
|---|---|
| 0 | check |
| 1 | No check |

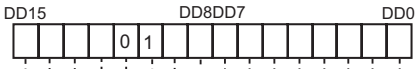
10. Address $09_{16}$, $10_{16}$, $17_{16}$ (=DA5 to 0)

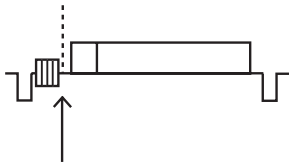



| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SEKI0 | Data slicer control bit 1 | SEKI1 SEKI0 N: 0 0 5 (Note1); 0 1 4 (Note2); 1 0 6 (Note3); 1 1 8 (Note4) | ○ | ○ |
| SEKI1 | | N-times the digital value after SEKI7.6. | ○ | ○ |
| SEKI2 | Data slicer control bit 2 | SEKI3 SEKI2 N: 0 0 4; 0 1 3; 1 0 1; 1 1 No differentiation | ○ | ○ |
| SEKI3 | | It differentiates from the digitized data in front of N/8 cycles (clock run-in cycle) to the digital value after SEKI0 and 1. | ○ | ○ |
| SEKI4 | Data slicer control bit 3 | SEKI5 SEKI4 N: 0 0 4; 0 1 3; 1 0 1; 1 1 No differentiation | ○ | ○ |
| SEKI5 | | It differentiates from the digitized data in after N/8 cycles (clock run-in cycle) to the digital value after SEKI3 and 2. | ○ | ○ |
| SEKI6 | Data slicer control bit 4 | SEKI6: 0 Average for four clocks; 1 It doesn't level (Leveling existence of the following A-D convert value) | ○ | ○ |
| SEKI7 | Data slicer control bit 5 | 0 The differentiation by SEKI2, 3, SEKI4, and 5 is differentiated by one time the value.<br>1 The differentiation by SEKI2, 3, SEKI4, and 5 is differentiated by twice the value. | ○ | ○ |
| Nothing is assigned. | | | × | × |
| SISLVL0 | Slice level control bit | 0 The clock line average level is used<br>1 The slice level selection bit (0C16,1316, 1A16, and house number SLS7?0) is used. | ○ | ○ |
| SISLVL1 | Slice level measurement period selection bit | 0 2 cycles of Clock run-in<br>1 4 cycles of Clock run-in | ○ | ○ |
| BIFON | Data format selection bit | 0 Non Return Zero<br>1 Bi-phase type | ○ | ○ |
| Reserved bit | | Must set to "0." | × | ○ |
| Reserved bits | | Must set to "1." | × | ○ |
| Reserved bit | | Must set to "0." | × | ○ |

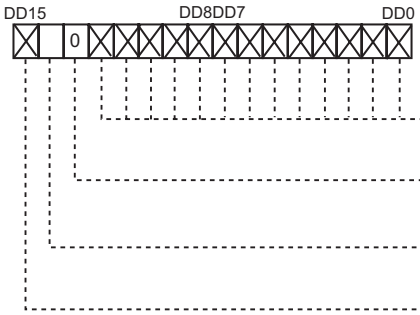
Note 1. When selecting 5 times with SEKI0, 1 and twice with SEKI7, select "none" for any one of SEKI2, 3 or SEKI4, 5.
Note 2. When selecting 8 times for SEKI0, 1, select "none" for both SEKI2, 3 and SEKI4, 5.
Note 3. When selecting 6 times for SEKI0, 1 and twice for SEKI7, select "none" for any one of SEKI2, 3 or SEKI4, 5.
Note 4. When selecting "none" for both SEKI2, 3 and SEKI4, 5, do not select 4 times, and select 8 times for SEKI0, 1.

11. Address $0A_{16}$, $11_{16}$, $18_{16}$ (=DA5 to 0)



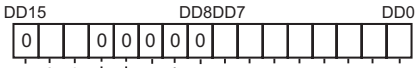

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SLS_HP0 | Slice check start position selection bit | It will become below if data slice start position is made into SLS_HS.<br>$SLS_HS = T2 \times \sum_{n=0}^{7} 2^n \, SLS_HPn$<br>T2 : Clock run-in cycle /2<br><br>The position where framing code begins to be checked is set up.<br>Setup in a 1-bit unit is possible. | ○ | ○ |
| SLS_HP1 | | | ○ | ○ |
| SLS_HP2 | | | ○ | ○ |
| SLS_HP3 | | | ○ | ○ |
| SLS_HP4 | | | ○ | ○ |
| SLS_HP5 | | | ○ | ○ |
| SLS_HP6 | | | ○ | ○ |
| SLS_HP7 | | | ○ | ○ |
| GET_HP0 | Phase fine-tuning bit | Slice data 0/1 judging clock is tuned finely. | ○ | ○ |
| GET_HP1 | | | ○ | ○ |
| Reserved bit | | Must set to "1." | 5 | ○ |
| Reserved bit | | Must set to "0." | 5 | ○ |
| GETPEEK0 | Peak detection period selection bit 0 | GETPEEK1 / GETPEEK0 / Clock run-in period:<br>0 / 0 / 2<br>0 / 1 / 4<br>1 / 0 / 5<br>1 / 1 / 6 | ○ | ○ |
| GETPEEK1 | Peak detection period selection bit 1 | | ○ | ○ |
| GETPEEK2 | Peak detection period selection bit 2 | 0: With clock compensation<br>1: With no clock compensation | ○ | ○ |
| GETPEEK3 | Peak detection period selection bit 3 | 0: Only a mountain is detected.<br>1: A mountain and a valley are detected. | ○ | ○ |

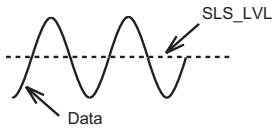
12. Address $0B_{16}$, $12_{16}$, $19_{16}$ (=DA5 to 0)



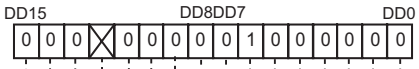

| Bit symbol | Bit name | Function | | R | W |
|---|---|---|---|---|---|
| Nothing is assigned. | | | | X | X |
| Reserved bit | | Must set to "0." | | X | ○ |
| FRAM | The number selection bit of framing code check bits | 0 | 15-bit check | ○ | ○ |
| | | 1 | 16-bit check | | |
| Nothing is assigned. | | | | X | X |

13. Address $0C_{16}$, $13_{16}$, $1A_{16}$ (=DA5 to 0)



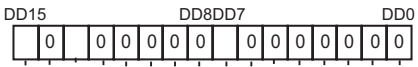

| Bit symbol | Bit name | Function | | R | W |
|---|---|---|---|---|---|
| SLS0 | Slice level selection bit | It will become below if a slice level is made into SLS_LVL.<br><br><br>At the time of SLS7 = "H"<br>$SLS_LVL = \sum_{n=0}^{6} 2^n SLSn - 128$<br>At the time of SLS7 = "L"<br>$SLS_LVL = \sum_{n=0}^{6} 2^n SLSn$ | | ○ | ○ |
| SLS1 | | | | ○ | ○ |
| SLS2 | | | | ○ | ○ |
| SLS3 | | | | ○ | ○ |
| SLS4 | | | | ○ | ○ |
| SLS5 | | | | ○ | ○ |
| SLS6 | | | | ○ | ○ |
| SLS7 | | | | ○ | ○ |
| Reserved bits | | Must set to "0." | | × | ○ |
| SELSTART | Slice start condition selection bit | 0 | Slice beginning after slice check period (SLS_HP7 to 0 of addresses 0A,11 and 18) passes | ○ | ○ |
| | | 1 | Slice beginning after standing up of clock run-in after slice check period (SLS_HP7 to 0 of addresses 0A,11 and 18) passes | | |
| GSTTIM | Ghost correct control bit | 0 | Ghost correct OFF | ○ | ○ |
| | | 1 | Ghost correct ON | | |
| Reserved bit | | Must set to "0." | | × | ○ |

14. Address $1B_{16}$ (=DA5 to 0)



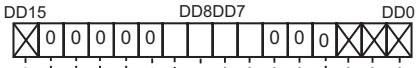

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Reserved bits | | Must set to "0." | × | ○ |
| Reserved bit | | Must set to "1." | × | ○ |
| Reserved bits | | Must set to "0." | × | ○ |
| Nothing is assigned. | | | × | × |
| Reserved bits | | Must set to "0." | × | ○ |

15. Address $1C_{16}$ (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Reserved bits | | Must set to "0." | X | ○ |
| Reserved bit | | Must set to "1" when EPG-J is acquired. Otherwise, set to "0." | X | ○ |
| Reserved bits | | Must set to "0." | X | ○ |
| EXT_PDC2 | Selection of PLL divided-in-3 frequency bit for PDC | 0 no divided<br>1 divided-in-3 | ○ | ○ |
| Reserved bit | | Must set to "0." | X | ○ |
| ADSTART | A/D conversion completion bit | 0 Conversion completion<br>1 Under conversion | ○ | ○ |

16. Address 1D16 (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Nothing is assigned. | | | × | × |
| Reserved bits | | Must set to "0." | × | ○ |
| PDC_VCO_ON | PDC clock oscillation selection bit | 0: PDC clock stop<br>1: PDC clock oscillation | ○ | ○ |
| PDC_VCO_R0<br>PDC_VCO_R1 | PDC clock oscillation change bit | PDC_VCO_R1 / PDC_VCO_R0: 0/0 Select PDC clock; 1/0 Select EPG-J clock; 0/1 Do not set up; 1/1 Do not set up | × | ○ |
| VPS_VCO_ON | VPS clock oscillation selection bit | 0: VPS clock stop<br>1: VPS clock oscillation | ○ | ○ |
| Reserved bits | | Must set to "0." | × | ○ |
| Nothing is assigned. | | | × | × |

17. Address 1E16 (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Reserved bits | | Must set to "0." | × | ○ |
| Nothing is assigned. | | | × | × |

18. Address 1F16 (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Nothing is assigned. | | | × | × |
| FLD1V | Field state flag | 0: Even field<br>1: Odd field | ○ | × |
| Reserved bits | | Must set to "0." | × | ○ |
| MACRO_ON | Synchronized signal search flag | 0: normal<br>1: unusual | ○ | × |
| Nothing is assigned. | | | × | × |

19. Address 2016 (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Reserved bits | | Must set to "0." | × | ○ |
| Nothing is assigned. | | | × | × |
| SEPV0 | Vertical synchronous separation standard selection bit | 0: Detected in L period of 15ms/22ms.<br>1: Detected in L period of 22ms. | ○ | ○ |
| Reserved bit | | Must set to "0." | × | ○ |
| NORMAL | Framing code check control bit | 0: Check (Data is acquired if Framing code is in agreement).<br>1: No check (All data is acquired). | ○ | ○ |
| LEVELA | Synchronous signal slice potential generating control bit | 0: Synchronous signal slice potential generating circuit OFF<br>1: Synchronous signal slice potential generating circuit ON | ○ | ○ |
| Reserved bits | | Must set to "0." | × | ○ |
| NXP | Broadcast method selection bit | NXP / MPAL / Broadcast method:<br>0 / 0 / NTSC<br>0 / 1 / M-PAL<br>1 / 0 / PAL<br>1 / 1 / Do not set up | ○ | ○ |
| MPAL | | | ○ | ○ |
| Reserved bit | | Must set to "0." | × | ○ |

20. Address $21_{16}$ (=DA5 to 0)



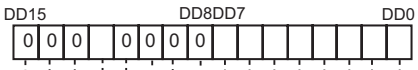

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Nothing is assigned. | | | × | × |
| Reserved bit | | Must set to "1." | × | ○ |
| Reserved bit | | Must set to "0." | × | ○ |

21. Address $22_{16}$ (=DA5 to 0)



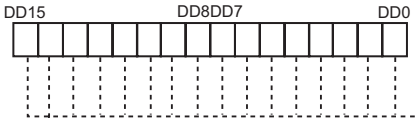

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DIV_PDCS0 | The PLL fine-tuning bit for PDC | Slice clock frequency $f_{PDC}$ for PDC is adjusted.<br>$f_{PDC} = f_{DIVP} \times \left( \sum_{n=0}^{8} 2^{n} \text{ DIV_PDCn} + \sum_{m=0}^{2} 2^{m-3} \text{ DIV_PDCSm+2} \right)$<br>$f_{DIVP}$ : Horizontal synchronized signal frequency<br>• When teletext (PDC) data is acquired (DIV_PDC4 to 0, DIV_PDCS2 to 0 = $(00100011)_2$)<br>• When EPG-J is acquired (DIV_PDC4 to 0, DIV_PDCS2 to 0 = $(00010011)_2$) | ○ | ○ |
| DIV_PDCS1 | | | ○ | ○ |
| DIV_PDCS2 | | | ○ | ○ |
| DIV_PDC0 | The divided value selection bit of PLL for PDC | | ○ | ○ |
| DIV_PDC1 | | | ○ | ○ |
| DIV_PDC2 | | | ○ | ○ |
| DIV_PDC3 | | | ○ | ○ |
| DIV_PDC4 | | | ○ | ○ |
| Nothing is assigned. | | | × | × |
| HM84SEL | 8/4 humming polarity selection bit | 0: Normal<br>1: The 4-bit data of 8/4 humming is reversal-outputted. | ○ | ○ |

22. Address $23_{16}$ (=DA5 to 0)



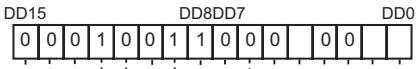

| Bit symbol | Bit name | Function | R | Ω |
|---|---|---|---|---|
| DIV_VPSS0 | The PLL fine-tuning bit for VPS | Slice clock frequency $f_{PDC}$ for VPS is adjusted. $f_{VPS} = f_{DIVV} \times \left( \sum_{n=0}^{8} 2^{n} \cdot DIV_VPSn + \sum_{m=0}^{2} 2^{m-3} \cdot DIV_VPSSm + 2 \right)$ $f_{DIVV}$ : Horizontal synchronized signal frequency When VPS is acquired (DIV_VPS4 to 0, DIV_VPSS2 to 0 = $(00100110)_2$) When CC, CC2X and ID-1 are acquired (DIV_VPS4 to 0, DIV_VPSS2 to 0 = $(11010100)_2$) | ○ | ○ |
| DIV_VPSS1 | | | ○ | ○ |
| DIV_VPSS2 | | | ○ | ○ |
| DIV_VPS0 | The divided value selection bit of PLL for VPS | | ○ | ○ |
| DIV_VPS1 | | | ○ | ○ |
| DIV_VPS2 | | | ○ | ○ |
| DIV_VPS3 | | | ○ | ○ |
| DIV_VPS4 | | | ○ | ○ |
| Reserved bits | | Must set to "0." | × | ○ |
| HORAX_ON | Horizontal synchronized signal selection bit | 0 Analog input<br>1 The digital input of HOR | ○ | ○ |
| Reserved bits | | Must set to "0." | × | ○ |

23. Address 24$_{16}$ (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Reserved bits | | Must set to "8808$_{16}$" when EPG-J is acquired. Otherwise, set to "0000$_{16}$." | × | ○ |

24. Address 25$_{16}$ (=DA5 to 0)



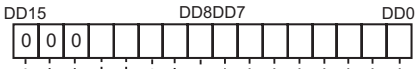

| Bit symbol | Bit name | Function | | R | W |
|---|---|---|---|---|---|
| ADSEL | A/D conversion slice bit | 0 | Normal | ○ | ○ |
| | | 1 | The digital value after A/D conversion is given from outside (with register). | | |
| ADON_TIM | A/D operation control bit | 0 | Programmable | ○ | ○ |
| | | 1 | Slice period | | |
| Reserved bits | | Must set to "0." | | ○ | ○ |
| SLICEON_TIM | Slice selection bit | 0 | Every line (CHECK_START) | ○ | ○ |
| | | 1 | Programmable (PRE_START) | | |
| Reserved bits | | Must set to "0." | | × | ○ |
| Reserved bits | | Must set to "1." | | × | ○ |
| Reserved bits | | Must set to "0." | | × | ○ |
| Reserved bit | | Must set to "1." | | × | ○ |
| Reserved bits | | Must set to "0." | | × | ○ |

25. Address $26_{16}$ (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DIVP_CK0 | The clock division value selection bit for phase comparison with a PDC clock | The divided clock used for the phase comparison with a PDC clock is set up.<br>$f_{SC} = f_{DIVP} \times \left( \sum_{n=0}^{7} 2^n \text{ DIVS_CKn} + 2 \right)$<br>$f_{DIVP}$: The slice clock frequency for PDC (please refer to DIV_PDCS0 to 2 and DIV_PDC0 to 4 (address $22_{16}$).)<br>When teletext (PDC) data is acquired<br>DIVP_CK7 to 0 = $(00001110)_2$<br>When EPG-J is acquired<br>DIVP_CK7 to 0 = $(00001001)_2$ | ○ | ○ |
| DIVP_CK1 | | | ○ | ○ |
| DIVP_CK2 | | | ○ | ○ |
| DIVP_CK3 | | | ○ | ○ |
| DIVP_CK4 | | | ○ | ○ |
| DIVP_CK5 | | | ○ | ○ |
| DIVP_CK6 | | | ○ | ○ |
| DIVP_CK7 | | | ○ | ○ |
| DIVV_CK0 | The clock division value selection bit for phase comparison with a VPS clock | The divided clock used for the phase comparison with a VPS clock is set up.<br>$f_{SC} = f_{DIVV} \times \left( \sum_{n=0}^{7} 2^n \text{ DIVV_CKn} + 2 \right)$<br>$f_{DIVV}$ : The slice clock frequency for VPS (refer to DIV_VPSS0 to 2 and DIV_VPS0 to 4 (address $23_{16}$).)<br>When VPS is acquired<br>DIVV_CK7 to 0 = $(00001110)_2$<br>When CC, CC2X and ID-1 are acquired<br>DIVV_CK7 to 0 = $(01010011)_2$ | ○ | ○ |
| DIVV_CK1 | | | ○ | ○ |
| DIVV_CK2 | | | ○ | ○ |
| DIVV_CK3 | | | ○ | ○ |
| DIVV_CK4 | | | ○ | ○ |
| DIVV_CK5 | | | ○ | ○ |
| DIVV_CK6 | | | ○ | ○ |
| DIVV_CK7 | | | ○ | ○ |

26. Address $27_{16}$ (=DA5 to 0)



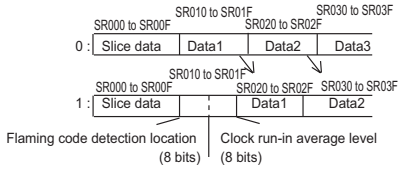

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DLYSEL0 | Data slicer control bit5 | These are the control bits of the ghost correction circuit. | ○ | ○ |
| DLYSEL1 | | | ○ | ○ |
| DLYSEL2 | | | ○ | ○ |
| DLYSEL3 | | | ○ | ○ |
| DLYSEL4 | | | ○ | ○ |
| DLYSEL5 | | | ○ | ○ |
| DLYSEL6 | | | ○ | ○ |
| DLYSEL7 | | | ○ | ○ |
| WEIGHT0 | Data slicer control bit6 | These are the control bits of the ghost correction circuit. | ○ | ○ |
| WEIGHT1 | | | ○ | ○ |
| WEIGHT2 | | | ○ | ○ |
| WEIGHT3 | | | ○ | ○ |
| WEIGHT4 | | | ○ | ○ |
| Reserved bits | | Must set to "0." | × | ○ |

27. Address $28_{16}$ (=DA5 to 0)

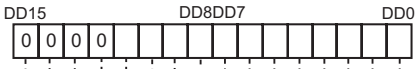



| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| ADLAT | Data acquisition selection bit | 0: Acquisition of slice data<br>1: Acquisition of A/D data | ○ | ○ |
| START | Slice data selection bit | Turning on the output after slice RAM of Flaming code detection position (8 bits) and the average of the clock run-in level (8 bits)and turning off are set (note.) | ○ | ○ |
| Reserved bit | | Must set to "0." | × | ○ |
| 6BITOFF | A/D lower bit selection bit | 0: Normal<br>1: Stop by 6th bit of A/D | ○ | ○ |
| Reserved bit | | Must set to "1" when EPG-J is acquired. Otherwise, set to "0." | × | ○ |
| SYNLVL0<br>SYNLVL1<br>SYNLVL2 | Synchronous signal slice level control bit | SYNLVL2 / SYNLVL1 / SYNLVL0 / Slice level<br>0 0 0 approx.1.10V±0.10V<br>0 0 1 approx.1.15V±0.10V<br>0 1 0 approx.1.20V±0.10V<br>0 1 1 approx.1.25V±0.10V<br>1 0 0 approx.1.30V±0.10V<br>1 0 1 approx.1.35V±0.10V<br>1 1 0 approx.1.40V±0.10V<br>1 1 1 approx.1.45V±0.10V | ○ | ○ |
| ADON | Data slicer control bit | 0: Data slicer OFF. (The amplifier for slicer is also turned off).<br>1: Data slicer ON (see INTAD and the INTDA about the amplifier for slicer) | ○ | ○ |
| INTAD | The amplifier control bit for data slicers | 0: Always data slicer ON.<br>1: On 3 to 23 lines and 315 to 335 line amplifier ON. On other line amplifier OFF | ○ | ○ |
| INTDA | The rudder resistance control bit for data slicers | 0: Always ladder resistance for data slicer ON.<br>1: On 3 to 23 lines and 315 to 335 line Ladder resistance ON. On other line Ladder resistance OFF | ○ | ○ |
| Reserved bits | | Must set to "0." | × | ○ |

Note. Slice RAM: Refer to Figure "Slice RAM bit construction"



28. Address $29_{16}$ (=DA5 to 0)



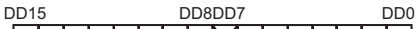

| Bit symbol | Bit name | Function | | R | W |
|---|---|---|---|---|---|
| VPS_VP0 | Setup of a slice start line (Shared by the first field and the second field) | If a slice start line is made into SLI_VS<br>At PAL<br><The first field><br>$SLI_VS = \sum_{n=0}^{8} 2^n VPS_VPn + 2$<br><the second field><br>$SLI_VS = \sum_{n=0}^{8} 2^n VPS_VPn + 315$<br>At NTSC<br><The first field><br>$SLI_VS = \sum_{n=0}^{8} 2^n VPS_VPn + 5$<br><the second field><br>$SLI_VS = \sum_{n=0}^{8} 2^n VPS_VPn + 268$<br>The data for 18 lines is stored in Slice RAM from the line set up by this register. | | ○ | ○ |
| VPS_VP1 | | | | ○ | ○ |
| VPS_VP2 | | | | ○ | ○ |
| VPS_VP3 | | | | ○ | ○ |
| VPS_VP4 | | | | ○ | ○ |
| VPS_VP5 | | | | ○ | ○ |
| VPS_VP6 | | | | ○ | ○ |
| VPS_VP7 | | | | ○ | ○ |
| VPS_VP8 | | | | ○ | ○ |
| SLI_GO | Slice ON/OFF control bit | 0 | Slice OFF | ○ | ○ |
| | | 1 | Slice ON | | |
| SYNCSEP_ON0 | Synchronous separate selection bit | 0 | Synchronous separate circuit OFF | ○ | ○ |
| | | 1 | Synchronous separate circuit ON | | |
| STBSYNCSEP | Synchronous separate input control bit | 0 | SYNCIN analog input | ○ | ○ |
| | | 1 | SYNCIN digital input | | |
| Reserved bits | | Must set to "0." | | × | ○ |

29. Address 2A16 (=DA5 to 0)



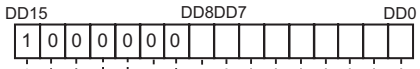

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| MASK0 | Mask width for time bases selection bit. | PAL 1135<br>NTSC 910<br>284<br>The position of mask release is set up 256 steps of setup can be performed in one fourth of the periods of the back between 1H<br>18H to 256H<br>Order of 0H to 17H<br>PAL<br>It cannot set up<br>NTSC 0H to 256H<br>Usually, please make it 80H | ○ | ○ |
| MASK1 | | | ○ | ○ |
| MASK2 | | | ○ | ○ |
| MASK3 | | | ○ | ○ |
| MASK4 | | | ○ | ○ |
| MASK5 | | | ○ | ○ |
| MASK6 | | | ○ | ○ |
| MASK7 | | | ○ | ○ |
| Reserved bits | | Must set to "0." | × | ○ |
| Reserved bit | | Must set to "1." | × | ○ |

30. Address 2B16 (=DA5 to 0)



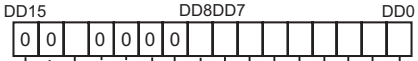

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Reserved bits | | Must set to "0." | × | ○ |
| Nothing is assigned. | | | × | × |
| Reserved bits | | Must set to "0." | × | ○ |
| SEL_PDCH | The internal H selection bit for data slicers | SEL_PDCH / SEL_VPSH: 0 0 External Hsync; 0 1 From PLL for VPS; 1 0 From PLL for PDC; 1 1 VPS or PDC | ○ | ○ |
| SEL_VPSH | | | ○ | ○ |
| Reserved bit | | Must set to "0." | × | ○ |
| SEL_PDEC | The clock selection bit for a PLL lock | 0 VPS and a PLL lock from Hsync.<br>1 VPS and a PLL lock from a X'tal system. | ○ | ○ |

31. Address 2C16 (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PLSPOS0 | Slice A/D ON period selection bit | Slice A/D ON period is counted.<br>V<br>H<br>H of $\left(\sum_{n=0}^{8} 2^n \text{ PLSPOSn}\right)$– th shot<br>H of $\left(\sum_{n=0}^{8} 2^n \text{ PLSNEGn}\right)$– th shot | ○ | ○ |
| PLSPOS1 | | | ○ | ○ |
| PLSPOS2 | | | ○ | ○ |
| PLSPOS3 | | | ○ | ○ |
| PLSPOS4 | | | ○ | ○ |
| PLSPOS5 | | | ○ | ○ |
| PLSPOS6 | | | ○ | ○ |
| PLSPOS7 | | | ○ | ○ |
| PLSPOS8 | | | ○ | ○ |
| Reserved bits | | Must set to "0." | × | ○ |
| Reserved bit | | Must set to "1." | × | ○ |

32. Address 2D16 (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PLSNEG0 | Slice-ON period selection bit | Refer to PLSPOS0 to 8 (Address 2C16) | ○ | ○ |
| PLSNEG1 | | | ○ | ○ |
| PLSNEG2 | | | ○ | ○ |
| PLSNEG3 | | | ○ | ○ |
| PLSNEG4 | | | ○ | ○ |
| PLSNEG5 | | | ○ | ○ |
| PLSNEG6 | | | ○ | ○ |
| PLSNEG7 | | | ○ | ○ |
| PLSNEG8 | | | ○ | ○ |
| Reserved bits | | Must set to "0." | × | ○ |
| Reserved bit | | Must set to "1" when teletext (PDC) data is acquired. | × | ○ |
| Reserved bits | | Must set to "0." | × | ○ |

33. Address 2E16 (=DA5 to 0)



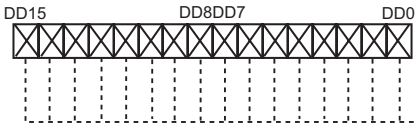

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| HCOUNT0 | Synchronous detection bit | A horizontal synchronized signal is counted. These bits are reset by set the VERTX bit (address 3316) to "0." | ○ | × |
| HCOUNT1 | | | ○ | × |
| HCOUNT2 | | | ○ | × |
| HCOUNT3 | | | ○ | × |
| HCOUNT4 | | | ○ | × |
| HCOUNT5 | | | ○ | × |
| HCOUNT6 | | | ○ | × |
| HCOUNT7 | | | ○ | × |
| HCOUNT8 | | | ○ | × |
| HCOUNT9 | | | ○ | × |
| HCOUNT10 | | | ○ | × |
| HCOUNT11 | | | ○ | × |
| HCOUNT12 | | | ○ | × |
| HCOUNT13 | | | ○ | × |
| HCOUNT14 | | | ○ | × |
| HCOUNT15 | | | ○ | × |

34. Address 2F16 (=DA5 to 0)



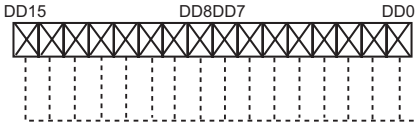

| Bit symbol | Bit name | Function | | R | W |
|---|---|---|---|---|---|
| Nothing is assigned | | | | × | × |
| Reserved bit | | Set to "0" usually | | × | ○ |
| STB_RES | Extended register all reset bit | 0 | Normal | ○ | ○ |
| | | 1 | It resets to address 0016 to the address 2E16 extended register. | | |

35. Address $30_{16}$ (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Nothing is assigned | | | × | × |

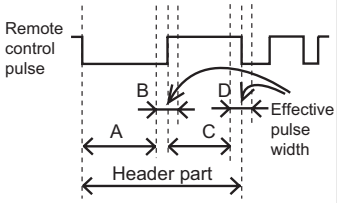
36. Address $31_{16}$ (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Nothing is assigned | | | × | × |

37. Address $32_{16}$ (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| RMHTD0(0) | Remote control header length selection bit | In order to detect a remote control pulse in standby mode, the header length to the oscillation for clocks (address $32_{16}$) is chosen.<br><br>$A = T_{XCIN} \times \sum_{n=0}^{8} 2^n$ RMHTD0(n)<br>$C = T_{XCIN} \times \sum_{n=0}^{8} 2^n$ RMHTD1(n)<br>$B = T_{XCIN} \times FILDIV0 \times \sum_{n=0}^{5}$ YUKOU0(n)<br>$D = T_{XCIN} \times FILDIV0 \times \sum_{n=0}^{5}$ YUKOU1(n)<br>$T_{XCIN}$ : XCIN pin input cycle<br>Division value set by FILDIV0 (bit 9 of address $33_{16}$) | ○ | ○ |
| RMHTD0(1) | | | ○ | ○ |
| RMHTD0(2) | | | ○ | ○ |
| RMHTD0(3) | | | ○ | ○ |
| RMHTD0(4) | | | ○ | ○ |
| RMHTD0(5) | | | ○ | ○ |
| RMHTD0(6) | | | ○ | ○ |
| RMHTD0(7) | | | ○ | ○ |
| RMHTD0(8) | | | ○ | ○ |
| JSTCKDIV0 | Clock division value of JUST CLOCK filter selection bit. | JSTCKDIV1 / JSTCKDIV0 / Main clock divided value<br>0 / 0 / 32 divided<br>0 / 1 / 64 divided<br>1 / 0 / 128 divided<br>1 / 1 / 256 divided | ○ | ○ |
| JSTCKDIV1 | | | ○ | ○ |
| JSTCKON | ON/OFF of JUST CLOCK filter selection bit. | 0 Filter OFF<br>1 Filter ON | ○ | ○ |
| Nothing is assigned. | | | × | × |
| RMTSEL | Remote control header polarity selection bit | 0 No reverse<br>1 reverse | ○ | ○ |

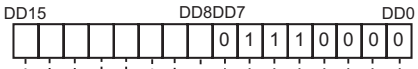
38. Address $33_{16}$ (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| RMHTD1(0) | Remote control header length selection bit | Refer to RMHTD0 (0) to (8) (address $32_{16}$). | ○ | ○ |
| RMHTD1(1) | | | ○ | ○ |
| RMHTD1(2) | | | ○ | ○ |
| RMHTD1(3) | | | ○ | ○ |
| RMHTD1(4) | | | ○ | ○ |
| RMHTD1(5) | | | ○ | ○ |
| RMHTD1(6) | | | ○ | ○ |
| RMHTD1(7) | | | ○ | ○ |
| RMHTD1(8) | | | ○ | ○ |
| FILDIV0 | Clock division value of remote control pulse selection bit | Clock division value for Remote control tolerance period measurement is selected. (Note 1)<br>FILDIV0: 0 = No divided, 1 = 2 | ○ | ○ |
| Reserved bit | | Must set to "0. " | ○ | ○ |
| FILON | Filter ON/OFF of remote control pulse selection bit (Notes 2, 3) | 0 OFF<br>1 ON | ○ | ○ |
| VERTX | Synchronous detection reset bit | 0 Reset<br>1 Horizontal synchronized signal count | ○ | ○ |
| Reserved bit | | Must set to "0. " | ○ | ○ |
| FILDIV1(0) | Filter of remote control clock divide value selection bit | FILDIV1(1) / FILDIV1(0) / Sub clock divided value:<br>0 / 0 / 2<br>0 / 1 / 4<br>1 / 0 / 8<br>1 / 1 / 16 | ○ | ○ |
| FILDIV1(1) | | | ○ | ○ |

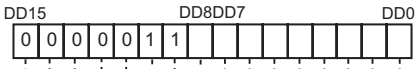
Notes 1. Refer to RMHTD0 (0) to (8) (address $32_{16}$)
Notes 2. Change these bits at initialization and do not rewrite during remote control receive.
Notes 3. The remote control pulse filter is only for the sub clock. Set it to OFF when the sub clock is not mounted.

39. Address $34_{16}$ (=DA5 to 0)

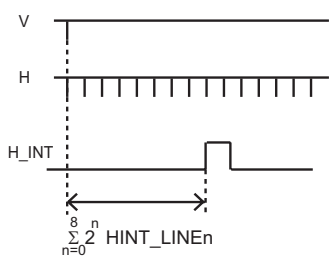



| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Reserved bits | | Must set to "$70_{16}$." | × | ○ |
| Reserved bits | | Must set to "$DD_{16}$" when EPG-J is acquired.<br>Otherwise, set to "$00_{16}$." | × | ○ |

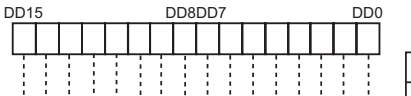
40. Address $35_{16}$ (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| HINT_LINE0 | H_INT interruption position selection bit | A period after V is inputted until H_INT rises is counted. | ○ | ○ |
| HINT_LINE1 | | | ○ | ○ |
| HINT_LINE2 | | | ○ | ○ |
| HINT_LINE3 | | | ○ | ○ |
| HINT_LINE4 | | | ○ | ○ |
| HINT_LINE5 | | | ○ | ○ |
| HINT_LINE6 | | | ○ | ○ |
| HINT_LINE7 | | | ○ | ○ |
| HINT_LINE8 | | | ○ | ○ |
| Reserved bits | | Must set to "1." | ○ | ○ |
| Reserved bits | | Must set to "0." | × | ○ |
| Reserved bit | | Must set to "0." | ○ | ○ |



41. Address $36_{16}$ (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| VINT0 | SLICEON interruption control test bit | 0000 : Interrupt disabled (Note 3)<br>1011 : Interrupt enabled<br>Others : Do not set up<br>When the period of data acquisition expires, the interrupt occurs by setting these bits to 1011.<br>Set up the TB5IC register (Note 4) when use by "Interrupt enabled." | ○ | ○ |
| VINT1 | | | ○ | ○ |
| VINT2 | | | ○ | ○ |
| VINT3 | | | ○ | ○ |
| INTRMT0 | Remote control interruption control bit (Note 1) | 0000 : Interrupt disabled (Note 3)<br>1010 : Interrupt enabled<br>Others : Do not set up<br>Set up the TB4IC register (Note 4) when use by "Interrupt enabled." | ○ | ○ |
| INTRMT1 | | | ○ | ○ |
| INTRMT2 | | | ○ | ○ |
| INTRMT3 | | | ○ | ○ |
| HINT0 | HINT interruption control test bit (Note 2) | 0000 : Interrupt disabled (Note 3)<br>1001 : Interrupt enabled<br>Others : Do not set up<br>Set up the TB3IC register (Note 4) when use by "Interrupt enabled." | ○ | ○ |
| HINT1 | | | ○ | ○ |
| HINT2 | | | ○ | ○ |
| HINT3 | | | ○ | ○ |
| SECINT0 | Clock timer interruption control bit | 0000 : Interrupt disabled (Note 3)<br>1000 : Interrupt enabled (Note 5)<br>Others : Do not set up<br>Set up the TB2IC register (Note 4) when use by "Interrupt enabled." | ○ | ○ |
| SECINT1 | | | ○ | ○ |
| SECINT2 | | | ○ | ○ |
| SECINT3 | | | ○ | ○ |

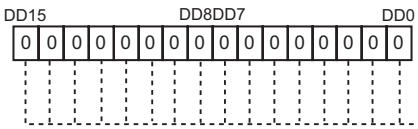
Notes 1. Refer to 15.6 Expansion Register Construction Composition.
Notes 2. Refer to the function of HINT_LINEn (Address $35_{16}$.)
Notes 3. Set these bits to 0000 when use the interrupt of Timer B2, Timer B3, Timer B4, or Timer B5.
Notes 4. Refer to Figure 6.3 Interrupt Control Registers.
Notes 5. When the second counter (Address $39_{16}$) is changed, an interrupt is generated every 1 second.
Notes 6. Please do not change data after setting initial data to address $36_{16}$ corresponding interrupt control bit VINTi, INTRMTi, HINTi, and SECINTi (i = 0 to 3) when you use the SLICEON, remote control, HINT, and the clock timer interrupt.

42. Address $37_{16}$ (=DA5 to 0)



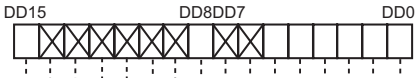

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| YUKOU0(0) | Remote control header judging pulse length selection bit 0 | Refer to RMTHD0(0) to (8) (Address $32_{16}$) | ○ | ○ |
| YUKOU0(1) | | | ○ | ○ |
| YUKOU0(2) | | | ○ | ○ |
| YUKOU0(3) | | | ○ | ○ |
| YUKOU0(4) | | | ○ | ○ |
| YUKOU0(5) | | | ○ | ○ |
| Nothing is assigned. | | | × | × |
| YUKOU1(0) | Remote control header judging pulse length selection bit 1 | Refer to RMTHD0(0) to (8) (Address $32_{16}$) | ○ | ○ |
| YUKOU1(1) | | | ○ | ○ |
| YUKOU1(2) | | | ○ | ○ |
| YUKOU1(3) | | | ○ | ○ |
| YUKOU1(4) | | | ○ | ○ |
| YUKOU1(5) | | | ○ | ○ |
| Nothing is assigned. | | | × | × |

43. Address $38_{16}$ (=DA5 to 0)\




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Reserved bit | | Must set to "0." | ○ | ○ |

44. Address $39_{16}$ (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SECOUT0 | Clock Timer Second Setting Bit | Set seconds (0 to 59 seconds) of clock timer.<br>The settable values are 0 to 59. | ○ | ○ |
| SECOUT1 | | | ○ | ○ |
| SECOUT2 | | | ○ | ○ |
| SECOUT3 | | | ○ | ○ |
| SECOUT4 | | | ○ | ○ |
| SECOUT5 | | | ○ | ○ |
| Nothing is assigned. | | | × | × |
| RTCON | Clock Timer Operation Selection Bit | 0 Clock timer stops<br>1 Clock timer operates | ○ | ○ |
| Nothing is assigned. | | | × | × |
| SECJUST | Second Just Setting Bit | When writing "1", less than second of the clock timer is reset.<br>When reading, the value is "0". | × | ○ |

45. Address $3A_{16}$ (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| MINOUT0 | Clock Timer Minute Setting Bit | Set hours and minutes of the clock timer by the minute.<br>The settable values are 0 to 1439 (00:00 to 23:59) | ○ | ○ |
| MINOUT1 | | | ○ | ○ |
| MINOUT2 | | | ○ | ○ |
| MINOUT3 | | | ○ | ○ |
| MINOUT4 | | | ○ | ○ |
| MINOUT5 | | | ○ | ○ |
| MINOUT6 | | | ○ | ○ |
| MINOUT7 | | | ○ | ○ |
| MINOUT8 | | | ○ | ○ |
| MINOUT9 | | | ○ | ○ |
| MINOUT10 | | | ○ | ○ |
| Nothing is assigned. | | | × | × |

46. Address 3B16 (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DAYCUONT0 | Clock Timer Day Setting Bit | Set days of the clock timer.<br>The settable value are 0 to 65535. | ○ | ○ |
| DAYCUONT1 | | | ○ | ○ |
| DAYCUONT2 | | | ○ | ○ |
| DAYCUONT3 | | | ○ | ○ |
| DAYCUONT4 | | | ○ | ○ |
| DAYCUONT5 | | | ○ | ○ |
| DAYCUONT6 | | | ○ | ○ |
| DAYCUONT7 | | | ○ | ○ |
| DAYCUONT8 | | | ○ | ○ |
| DAYCUONT9 | | | ○ | ○ |
| DAYCUONT10 | | | ○ | ○ |
| DAYCUONT11 | | | ○ | ○ |
| DAYCUONT12 | | | ○ | ○ |
| DAYCUONT13 | | | ○ | ○ |
| DAYCUONT14 | | | ○ | ○ |
| DAYCUONT15 | | | ○ | ○ |

47. Address 3C16 (=DA5 to 0) (Note 1)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| RMTSTART | Remote control transmission start bit | Start and the stop of wave form output is selected.<br>0: Stop<br>1: Start | ○ | ○ |
| IRPOL(0) | Carrier selection bit | The output mode of remote control transmission wave form is selected.<br>000: External wave, Carrier and AND output.<br>010: External wave, Carrier and OR output.<br>101: Only external wave is output (carrier none.)<br>Additionally: Setting prohibition. | ○ | ○ |
| IRPOL(1) | | | | |
| IRPOL(2) | | | | |
| RMCDIV(0) | Carrier clock source selection bit | Carrier's clock source is selected.<br>00: The main clock (There is not dividing frequency.)<br>01: 2.5 dividing of the main clock.<br>10: 8 dividing of the main clock.<br>11: Setting prohibition. | ○ | ○ |
| RMCDIV(1) | | | | |
| RMTDIV(0) | External wave clock source selection bit | The clock source of external wave is selected.<br>00: The main clock (There is not dividing frequency.)<br>01: 8 dividing of the main clock.<br>10: 64 dividing of the main clock.<br>11: 256 dividing of the main clock. | ○ | ○ |
| RMTDIV(1) | | | | |
| RMT_TM(0) | External wave clock dividing frequency value setting bit | Dividing frequency value which is selected by RMTDIV(1, 0)<br>Set RMT_TM.<br>$RMT_TM = \sum_{n=0}^{7} 2^n \cdot RMT_TM0(n)$ | ○ | ○ |
| RMT_TM(1) | | | | |
| RMT_TM(2) | | | | |
| RMT_TM(3) | | | | |
| RMT_TM(4) | | | | |
| RMT_TM(5) | | | | |
| RMT_TM(6) | | | | |
| RMT_TM(7) | | | | |

Note 1. Refer to 14.6 "(6) Remote control transmission function."

48. Address 3D16 (=DA5 to 0) (Note 1)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| RMT_TML(0) | Carrier L period setting bit | Carrier's L period is set.<br>$RMT_TML = \sum_{n=0}^{7} 2^n \cdot RMT_TML(n)$ | ○ | ○ |
| RMT_TML(1) | | | | |
| RMT_TML(2) | | | | |
| RMT_TML(3) | | | | |
| RMT_TML(4) | | | | |
| RMT_TML(5) | | | | |
| RMT_TML(6) | | | | |
| RMT_TML(7) | | | | |
| RMT_TMH(0) | Carrier H period setting bit | Carrier's H period is set.<br>$RMT_TMH = \sum_{n=0}^{7} 2^n \cdot RMT_TMH(n)$ | ○ | ○ |
| RMT_TMH(1) | | | | |
| RMT_TMH(2) | | | | |
| RMT_TMH(3) | | | | |
| RMT_TMH(4) | | | | |
| RMT_TMH(5) | | | | |
| RMT_TMH(6) | | | | |
| RMT_TMH(7) | | | | |

Note 1. Refer to 14.6 "(6) Remote control transmission function."

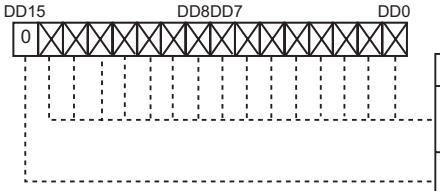
49. Address 3E16 (=DA5 to 0) (Note 1)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| RMTTXINT(0) | Remote control transmission interrupt control (Note 1, Note 4) | 0000: Interrupt prohibition (Note 2)<br>0001: Interrupt permission<br>Additionally: Setting prohibition<br>Please set INT2IC register when using it by "Interrupt permission." | ○ | ○ |
| RMTTXINT(1) | | | | |
| RMTTXINT(2) | | | | |
| RMTTXINT(3) | | | | |
| Reserved bit | | Must set to "0." | × | ○ |
| Nothing is assigned. | | | × | × |
| IROUT_SLICEON | P84 output signal selection bit 2 (Note 3) | 0: Normal setting (When use P84 I/O port or INT2 interrupt.)<br>1: Remote control transmission pulse output | ○ | ○ |
| Reserved bit | | Must set to "0." | × | ○ |

Notes 1. Refer to 14.6 "(6) Remote control transmission function."
Notes 2. Set and use 00002 when use INT2 interrupt.
Notes 3. Set "1" to bit b4 of port P8 direction register when use the remote control transmission function.
Notes 4. Please do not change data after setting the initialization data to remote control transmission interrupt control bit RMTTXINT(i) (i = 0 to 3) when you use the remote control transmission interrupt.

50. Address $3F_{16}$ (=DA5 to 0) (Note 1)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Nothing is assigned. | | | X | X |
| Reserved bit | | Must set to "0." | X | O |

Note 1. Refer to 14.6 "(6) Remote control transmission function."

51. Address $40_{16}$ (=DA5 to 0)




| Bit symbol | Bit mane | Function | R | W |
|---|---|---|---|---|
| FPLS_MIN0 | Fixed length pulse lower bound value setting bit | The fixed length pulse lower bound value is set. (note 1) | O | O |
| FPLS_MIN1 | | | O | O |
| FPLS_MIN2 | | | O | O |
| FPLS_MIN3 | | | O | O |
| FPLS_MIN4 | | | O | O |
| FPLS_MIN5 | | | O | O |
| FPLS_MIN6 | | | O | O |
| FPLS_MIN7 | | | O | O |
| FPLS_MAX0 | Fixed length pulse upper bound value setting bit | The fixed length pulse upper bound value is set. (note 1) | O | O |
| FPLS_MAX1 | | | O | O |
| FPLS_MAX2 | | | O | O |
| FPLS_MAX3 | | | O | O |
| FPLS_MAX4 | | | O | O |
| FPLS_MAX5 | | | O | O |
| FPLS_MAX6 | | | O | O |
| FPLS_MAX7 | | | O | O |

*Note 1: The fixed value pulse is a pulse ("H" or "L" part) where 0/1 is not judged. It does according to enhancing register VBITPOL (Address $43_{16}$ and bit 13) at H period of 0/1 judgments or it does at L period or it selects it.

52. Address $41_{16}$ (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DT0_MIN0 | "0" data pulse length lower bound value setting bit | "0" data pulse length lower bound value is set. | ○ | ○ |
| DT0_MIN1 | | | ○ | ○ |
| DT0_MIN2 | | | ○ | ○ |
| DT0_MIN3 | | | ○ | ○ |
| DT0_MIN4 | | | ○ | ○ |
| DT0_MIN5 | | | ○ | ○ |
| DT0_MIN6 | | | ○ | ○ |
| DT0_MIN7 | | | ○ | ○ |
| DT0_MAX0 | "0" data pulse elder limit value setting bit | "0" data pulse elder limit value is set | ○ | ○ |
| DT0_MAX1 | | | ○ | ○ |
| DT0_MAX2 | | | ○ | ○ |
| DT0_MAX3 | | | ○ | ○ |
| DT0_MAX4 | | | ○ | ○ |
| DT0_MAX5 | | | ○ | ○ |
| DT0_MAX6 | | | ○ | ○ |
| DT0_MAX7 | | | ○ | ○ |

53. Address $42_{16}$ (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DT1_MIN0 | "1" data pulse length lower bound value setting bit | "1" data pulse length lower bound value is set. | ○ | ○ |
| DT1_MIN1 | | | ○ | ○ |
| DT1_MIN2 | | | ○ | ○ |
| DT1_MIN3 | | | ○ | ○ |
| DT1_MIN4 | | | ○ | ○ |
| DT1_MIN5 | | | ○ | ○ |
| DT1_MIN6 | | | ○ | ○ |
| DT1_MIN7 | | | ○ | ○ |
| DT1_MAX0 | "1" data pulse elder limit value setting bit | "1" data pulse elder limit value is set. | ○ | ○ |
| DT1_MAX1 | | | ○ | ○ |
| DT1_MAX2 | | | ○ | ○ |
| DT1_MAX3 | | | ○ | ○ |
| DT1_MAX4 | | | ○ | ○ |
| DT1_MAX5 | | | ○ | ○ |
| DT1_MAX6 | | | ○ | ○ |
| DT1_MAX7 | | | ○ | ○ |

54. Address $43_{16}$ (=DA5 to 0)



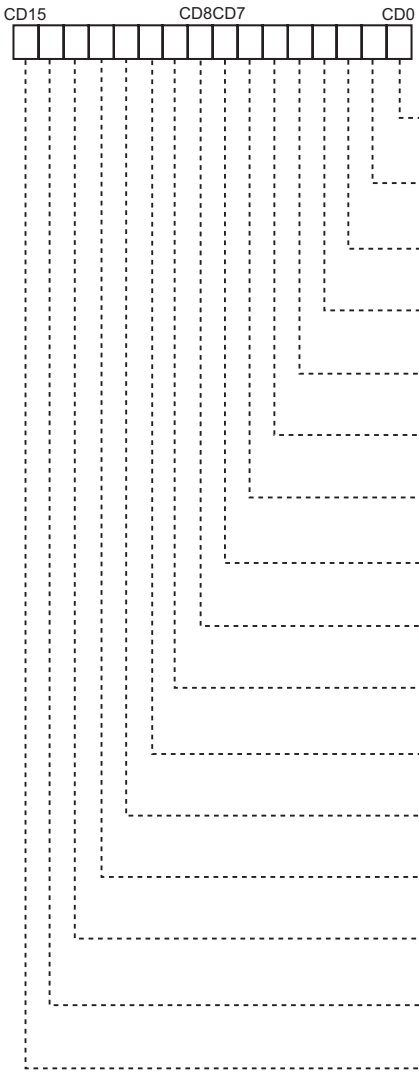

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| MAXBIT0 | The maximum reception bit number setting bit | The number of maximum reception bits is set. | ○ | ○ |
| MAXBIT1 | | | ○ | ○ |
| MAXBIT2 | | | ○ | ○ |
| MAXBIT3 | | | ○ | ○ |
| MAXBIT4 | | | ○ | ○ |
| MAXBIT5 | | | ○ | ○ |
| MINBIT0 | Minimum reception bit number setting bit | The number of minimum reception bits is set. | ○ | ○ |
| MINBIT1 | | | ○ | ○ |
| MINBIT2 | | | ○ | ○ |
| MINBIT3 | | | ○ | ○ |
| MINBIT4 | | | ○ | ○ |
| MINBIT5 | | | ○ | ○ |
| RMTLSB | LSB/MSB reception selection bit | 0:It receives it with MSB.<br>1:It receives it with LSB. | ○ | ○ |
| VBITPOL | 0/1 judgment level selection bit | 0:0/1 is judged at L period.<br>1:0/1 is judged at H period. | ○ | ○ |
| INTSEL | Interrupt selection bit | 0:After it matches it, interrupt is generated.<br>1:When a reception of the data of the 16th bit and the maximum data are received, interrupt is generated | ○ | ○ |
| GET_DATA | Bit that takes data | 1:When "1" is written from the reception buffer in the reception register, the data of the reception buffer is written in the reception register.<br>(The value when reading it is 0. ) | × | ○ |

55. Address $44_{16}$ (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DATAOL0 | Number of receive data bits | The number of bits of received data is displayed. | ○ | × |
| DATAOL1 | | | | |
| DATAOL2 | | | | |
| DATAOL3 | | | | |
| DATAOL4 | | | | |
| DATAOL5 | | | | |
| NU0 | Number of receive data words | 16 piece nbit data was received. (n=0,1,2) | ○ | × |
| NU1 | | | | |
| INTRMTFL0 | One word reception completion flag | The reception completion of 0/1 data for one word (16 bits) is shown.<br>0:The data of the 16th bit is not received.<br>1:The data of the 16th bit was received | ○ | × |
| INTRMTFL1 | Flag that received data of number of maximum bits | 0:The data of the number of maximum bits was not received.<br>1:The data of the number of maximum bits was received | ○ | × |
| INTRMTFL2 | Flag that received the data everything | 0:Everything did not receive the data.<br>1:Everything received the data. | ○ | × |
| NG | NG flag bit of reception | 1:There was NG while receiving it. | ○ | × |
| Nothing is assigned. | | | × | × |

56. Address 45$_{16}$ (=DA5 to 0)




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| RMTDA0 | Reception register | The received data ("H" data 0/1 judgments by the period of "L") is stored. | ○ | × |
| RMTDA1 | | | ○ | × |
| RMTDA2 | | | ○ | × |
| RMTDA3 | | | ○ | × |
| RMTDA4 | | | ○ | × |
| RMTDA5 | | | ○ | × |
| RMTDA6 | | | ○ | × |
| RMTDA7 | | | ○ | × |
| RMTDA8 | | | ○ | × |
| RMTDA9 | | | ○ | × |
| RMTDA10 | | | ○ | × |
| RMTDA11 | | | ○ | × |
| RMTDA12 | | | ○ | × |
| RMTDA13 | | | ○ | × |
| RMTDA14 | | | ○ | × |
| RMTDA15 | | | ○ | × |

## 14.6 Expansion Register Construction Composition

### (1) Acquisition timing

The SLICEON signal is output in the acquisition possible period.

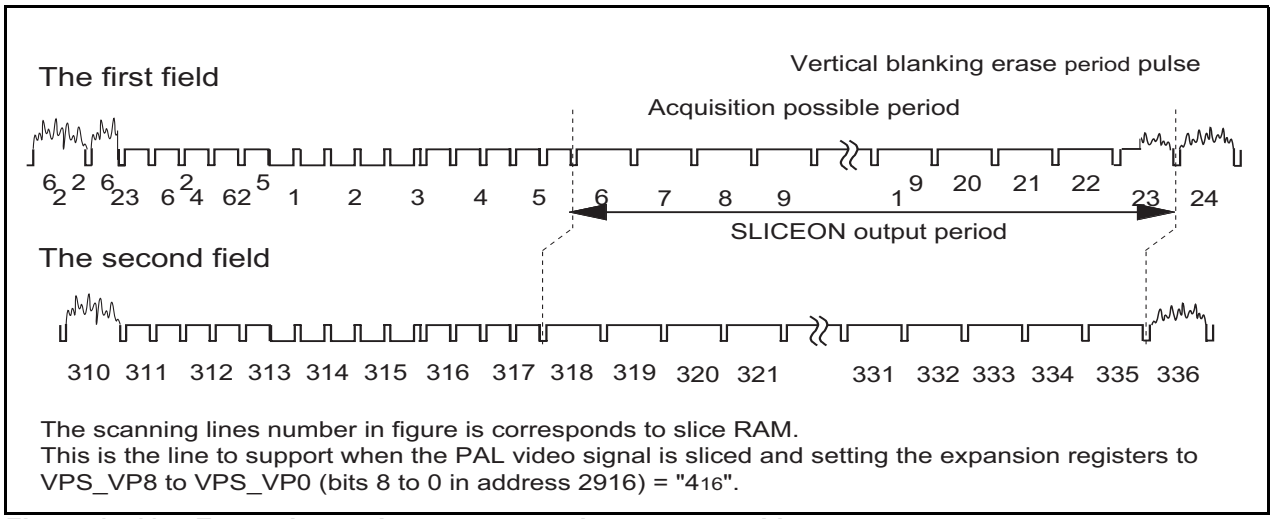



**Figure 14.11 Expansion register access registers composition**

### (2) Synchronized signal detection circuit

The number of pulses of the horizontal synchronized signal of a compound video signal is counted during a fixed period. The horizontal synchronous number of pulses can always be read from an expansion register.
A block diagram is shown in Figure. 14.12.

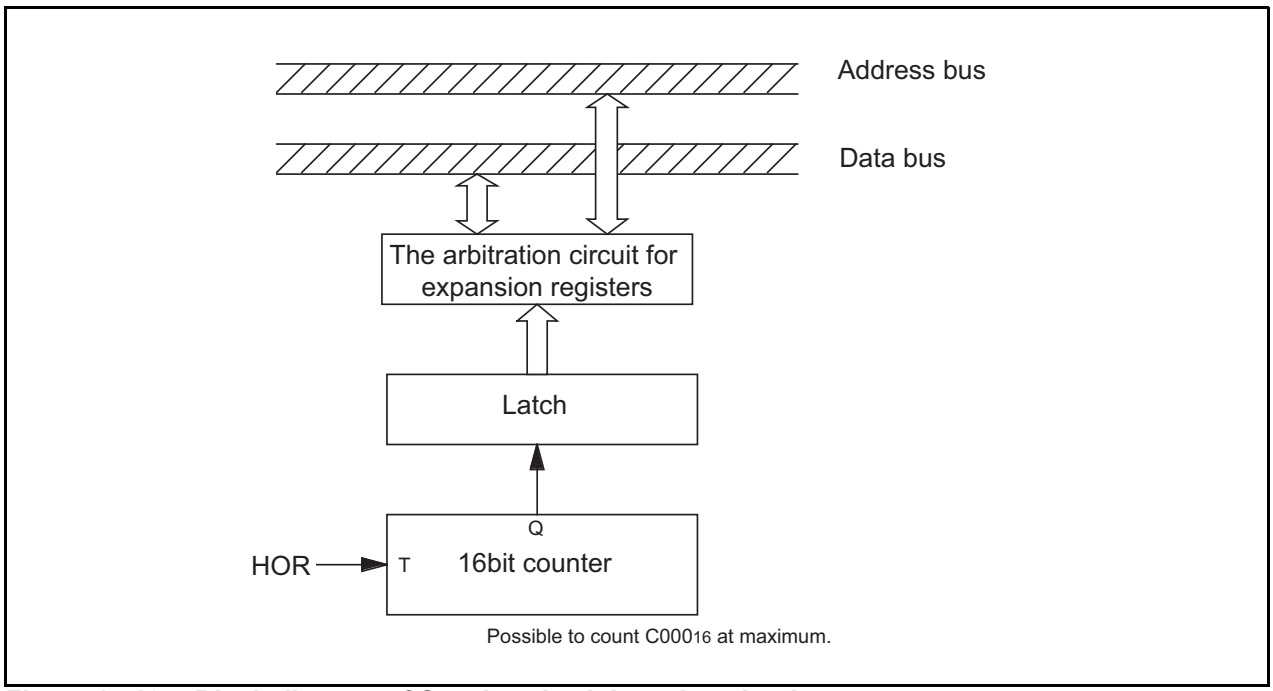



**Figure 14.12 Block diagram of Synchronized detection circuit**

### (3) Register related to Slicer

The relation between V, H signal, and the register related to slicer is shown in Figure. 14.13 and Figure. 14.14.

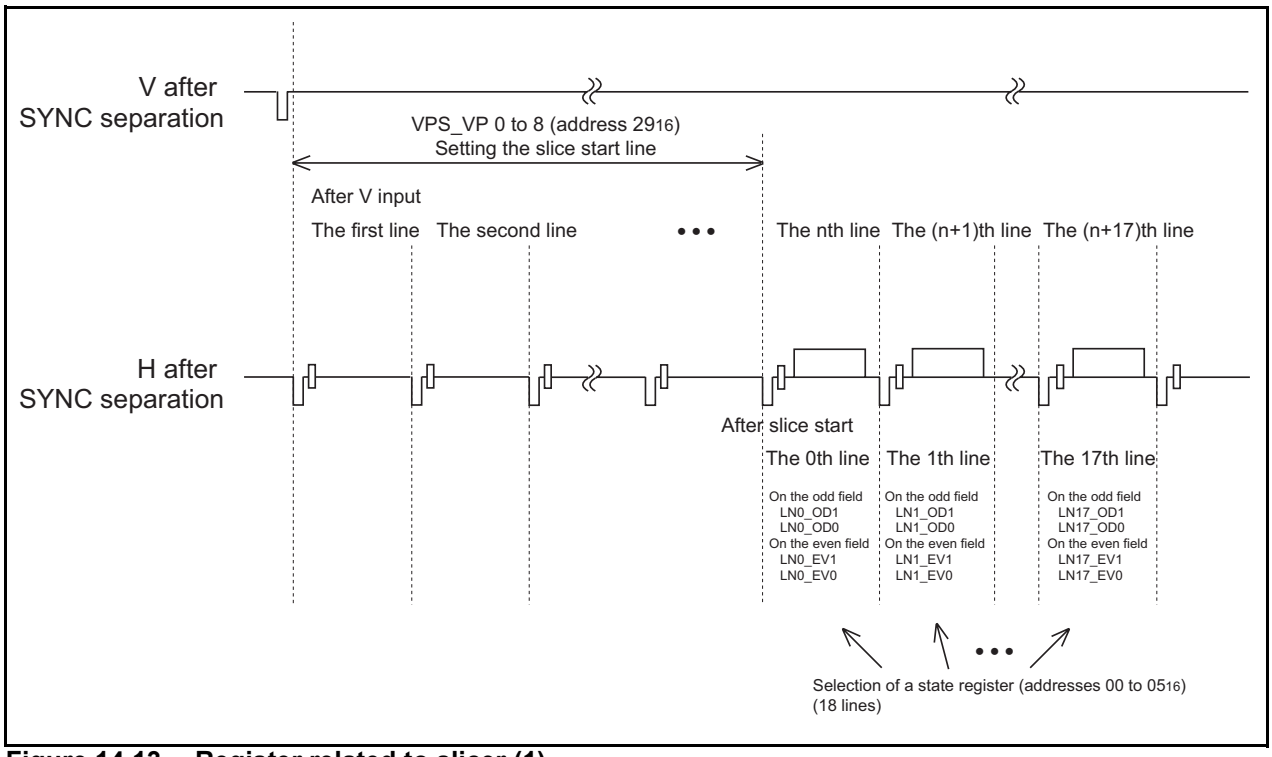



**Figure 14.13 Register related to slicer (1)**

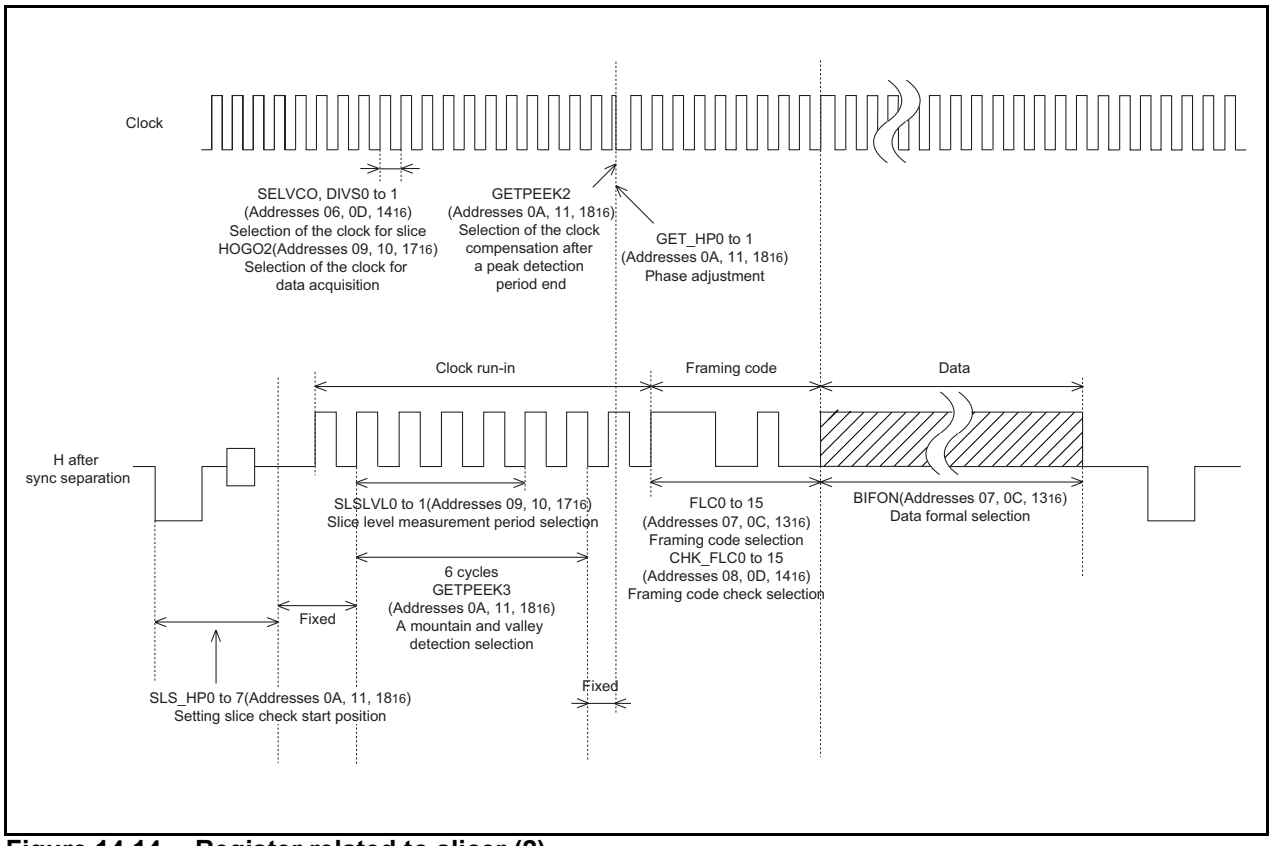



**Figure 14.14 Register related to slicer (2)**

### (4) Remote control pattern recognition

Pattern matching of remote control is performed using a sub clock oscillation. Remote control input is input from RMTIN terminal. Interruption is generated when pattern matching is in agreement.

4 times match noise filter is being included, in front of the pattern matching circuit. A block diagram is shown in Figure 14.15.

The example of a waveform of pattern matching is shown in Figure.14.16. The flow of pattern matching is shown in Figure.14.17.

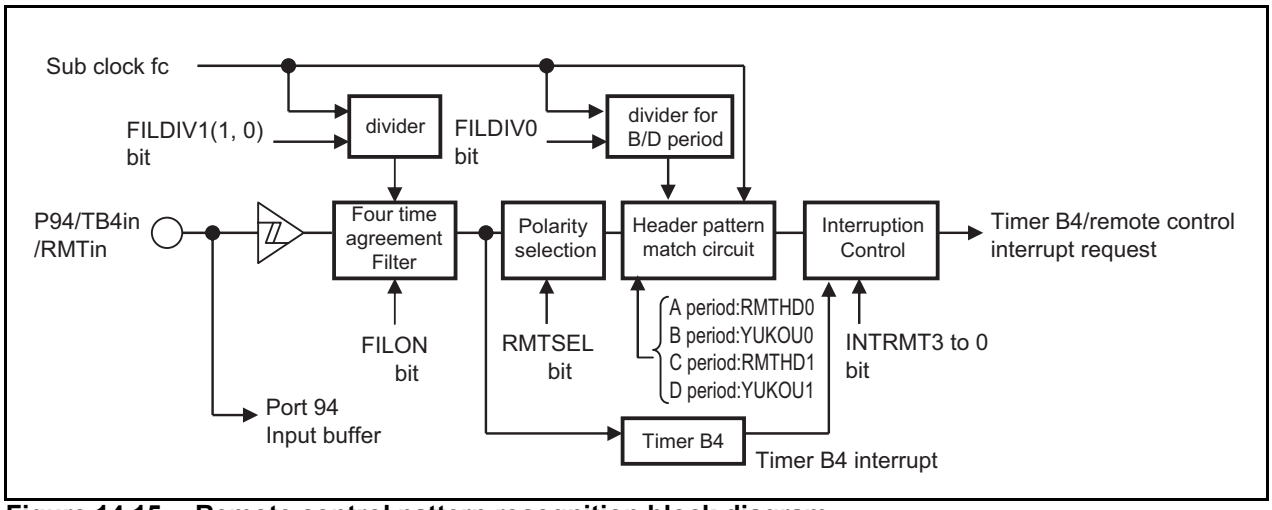
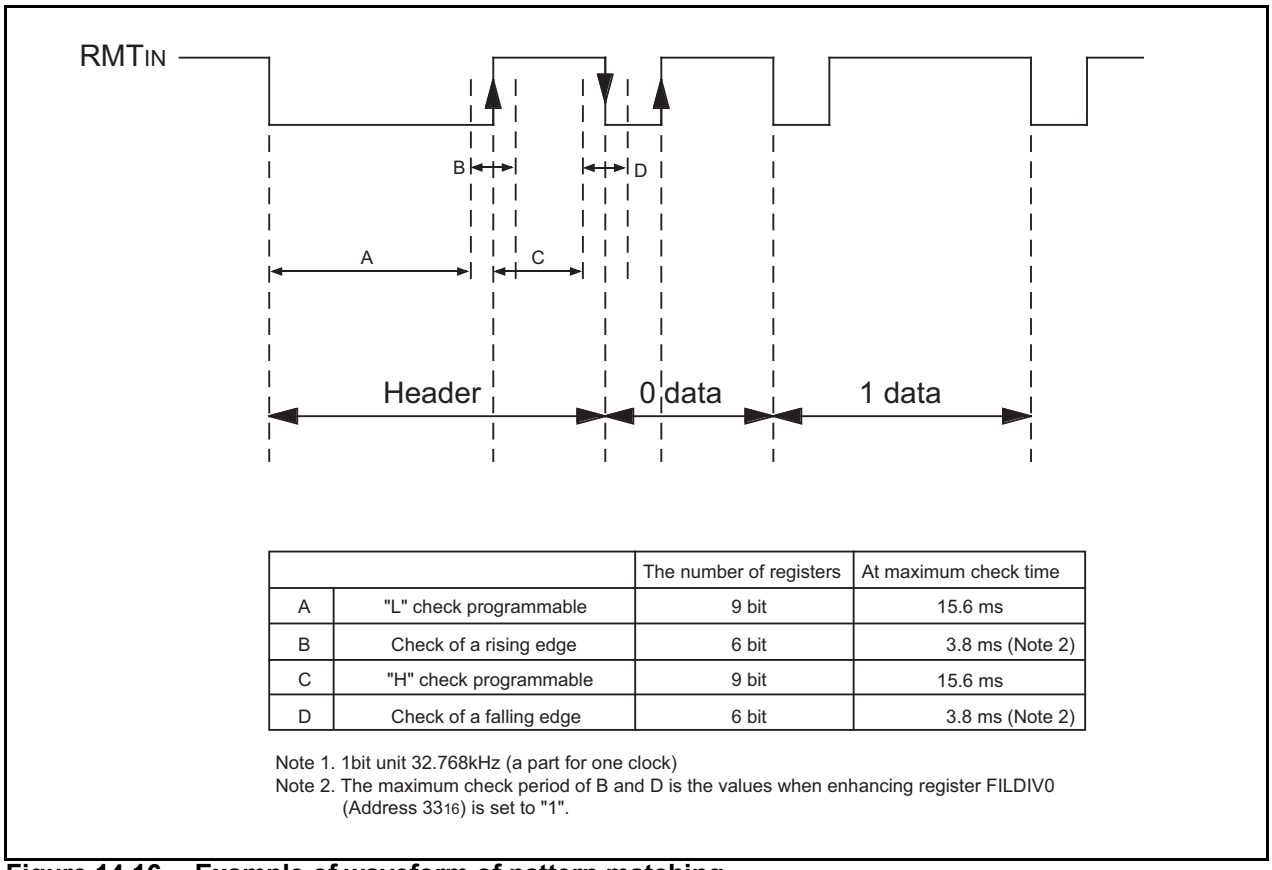



**Figure 14.15 Remote control pattern recognition block diagram**




| | | The number of registers | At maximum check time |
|---|---|---|---|
| A | "L" check programmable | 9 bit | 15.6 ms |
| B | Check of a rising edge | 6 bit | 3.8 ms (Note 2) |
| C | "H" check programmable | 9 bit | 15.6 ms |
| D | Check of a falling edge | 6 bit | 3.8 ms (Note 2) |

Note 1. 1bit unit 32.768kHz (a part for one clock)
Note 2. The maximum check period of B and D is the values when enhancing register FILDIV0 (Address $33_{16}$) is set to "1".

**Figure 14.16 Example of waveform of pattern matching**

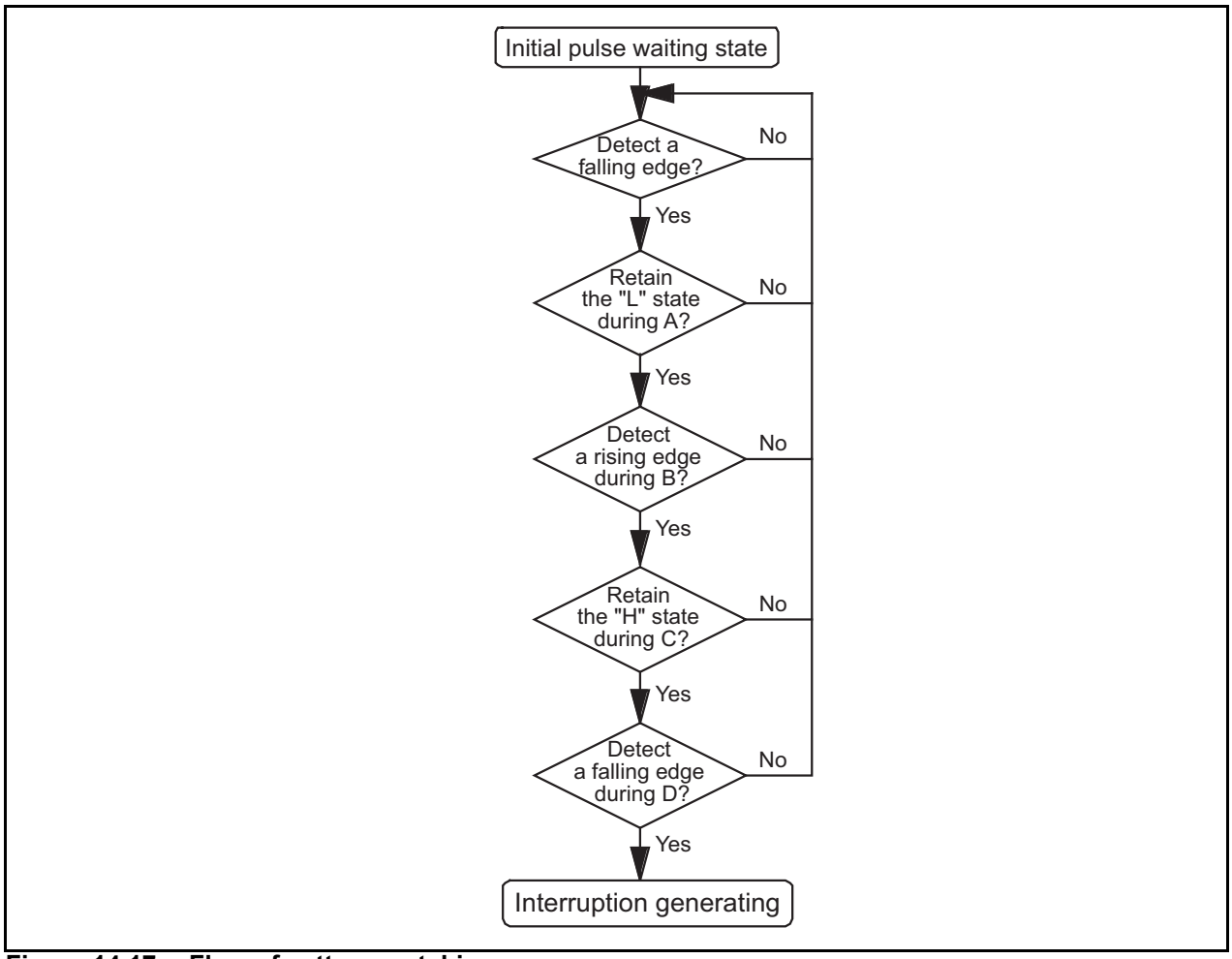



**Figure 14.17 Flow of pattern matching**

## 4 times match process operation

This is a 4 times match digital filter using the sub clock oscillation. The input signal of the RMTin pin is sampled 4 times and the output level will change only when the level matches 4 times. When using this filter, set the FILON bit (bit11) in the expansion register 33H to "1". The signal after passing 4 times match filter is applied to the header pattern matching circuit and timer B4 at filter ON. The sampling rate can be changed by the FILDIV1 (1, 0) bit in the expansion register 33H. Refer to the FILDIV1 bit of the expansion register 33H function description for details. The input signal is through supplied to the latter circuit at filter OFF. (no clock delay).Since this filter operates only with the sub clock and cannot be used for the main clock, set the filter to OFF (FILON bit ="0") when the sub clock is not mounted.

### (5) Clock timer function

The sub clock is selected as the count source and the clock timer can set and read the count value every day, minute and second. It has the following functions.

**Clock function**

1.This timer is dedicated clock function which is independent from timer A and timer B.
2.The settable ranges of day, minute and second are 0 to 65535 days, 0 to 1439 minutes and 0 to 59 seconds.
3.Second just setting is available (reset the count value of less than second).

**1 second interrupt**

1.The interrupt request is generated when second of the clock timer is incremented.
(The interrupt request is not generated at the second just setting.)

### (6) Remote control transmission function

The career is synthesized to two kinds of pulses of external wave, and the remote control transmission circuit is output from the microcomputer pin.

The specification of the remote control transmitter is shown in Table 14.5, The remote control transmission circuit output wave form is shown in Figure 14.18, and The remote control transmission circuit block chart is shown in Figure 14.19.

The feature is shown below.

- Career is a continuous pulse of the arbitrary width obtained by dividing the main clock (Figure 14.18 Wave forms 2 and 3.)
- External wave is a shape of waves generated from the transmission data buffer with reading pin output value ("H"/"L") and the pulse width one by one (Figure 14.18 Wave form 4.)
- The shape of waves that can synthesize the career to external wave is output from the pin (Figure 14.18 Wave form 5.)

**Table 14.5 Specification of remote control transmission function**

| Item | Specifications |
|---|---|
| Count source | Carrier: Selection from f1 (There is not XIN dividing frequency), f2.5 (XIN 2.5 dividing frequency), and f8 (XIN 8 dividing frequency.)<br>External wave: Selection from f1 (There is not XIN dividing frequency), f8 (XIN 8 dividing frequency), f64 (XIN 64 dividing frequency), and f256 (XIN 256 dividing frequency.) |
| Count operation (carrier) | ● Down count<br>● The register for “H” width setting is read by standing up about the pulse, and it continues the count.<br>● The register for “L” width setting is read by standing up about the pulse, and it continues the count. |
| Comparing of dividing frequency (carrier) | ● “H” period and “L” period are 1 to 256. |
| Count operation (external wave) | ● Down count<br>● Wave form output value (“H”/“L”) and the count value are read from the remote control transmission data buffer with stand up/fall down of the pulse, and the port output and the count are continued. |
| Comparing of dividing frequency (external wave) | 1 to 16384 (14bit) |
| Start of counting condition | The remote control transmission start bit is set to “1.” |
| Count stop condition | ● The remote control transmission start bit is set to “0.”<br>● After the untransmission data number reference bit is empty and the count value is underflow |
| Interrupt generation timing | ● When external wave stand up/fall down (When the value of the interrupt setting bit read from the remote control transmission data buffer is only 1. ) |
| Remote control transmission mode | ● OR output of career and external wave<br>● Only external wave is output (carrier none.) |

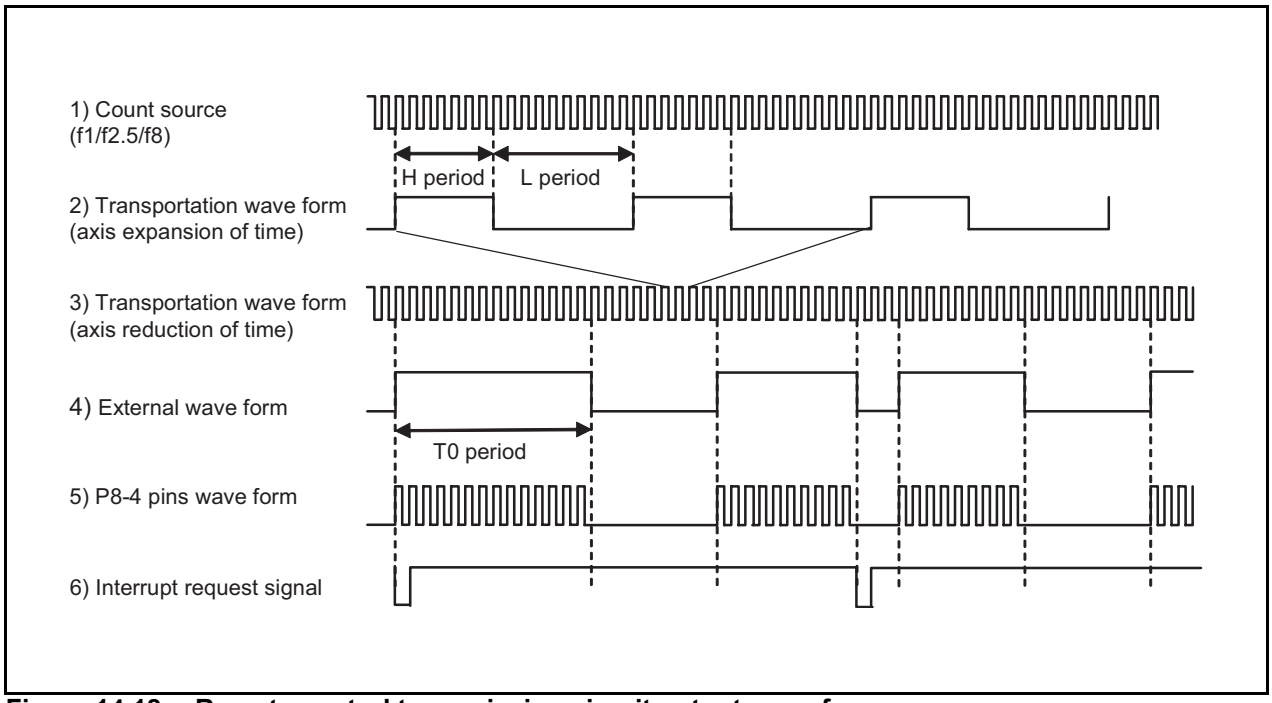



**Figure 14.18 Remote control transmission circuit output wave form**

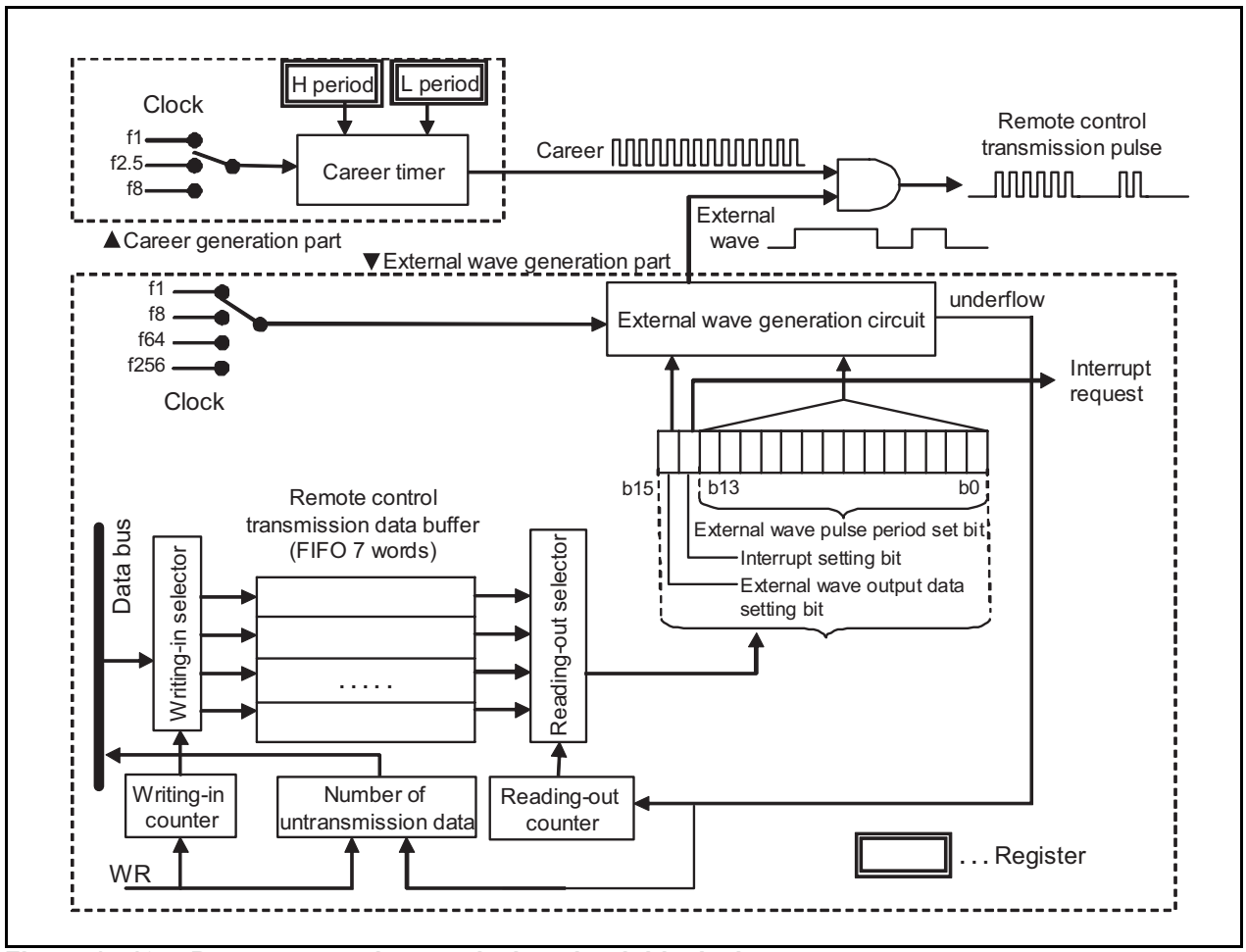
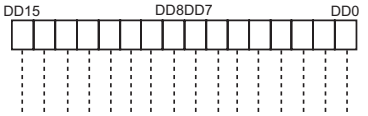



**Figure 14.19 Remote control transmission circuit block chart**

Remote control transmission data buffer register




| Symbol | Address | When reset |
|---|---|---|
| RMT_TMHL | $200_{16}$ | $0000_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| RMT_TMHL (0) | External wave pulse period setting/Untransmission data number reference bit | When writing in: Set one pulse period of the remote control transmission external wave.<br>When reading out: The number of data (number of untransmission data) that remains in the remote control transmission data buffer is read out. | ○ | ○ |
| RMT_TMHL (1) | | | | |
| RMT_TMHL (2) | | | | |
| RMT_TMHL (3) | | | | |
| RMT_TMHL (4) | | | | |
| RMT_TMHL (5) | | | | |
| RMT_TMHL (6) | | | | |
| RMT_TMHL (7) | | | | |
| RMT_TMHL (8) | | | | |
| RMT_TMHL (9) | | | | |
| RMT_TMHL (10) | | | | |
| RMT_TMHL (11) | | | | |
| RMT_TMHL (12) | | | | |
| RMT_TMHL (13) | | | | |
| RMT_TMHL (14) | Interrupt setting bit | The presence of the interrupt generation is specified at the change of the external wave output.<br>0:The interrupt request signal is not generated.<br>1:The interrupt request signal is generated. | ○ | ○ |
| RMT_TMHL (15) | External wave output data setting bit | The remote control transmission external wave data is set. | ○ | ○ |

**Figure 14.20 Setting of enhancing register (Only the part related to the remote control transmission.)**

## 14.7 8/4 Humming Decoder

8/4 humming decoder operates only by written the data which is 8/4 humming- encoded to 8/4 humming register (address 021A16). 8/4 humming register consists of 16 bits, can decode two data at once. Can obtain the decoded result by reading 8/4 humming register, and the decoded value and error information are output. Corrects and outputs the decoded value for single error, and outputs only error information for double error. Decoded result is shown in Figure 14.21 and humming 8/4 register composition is shown in Figure 14.22.

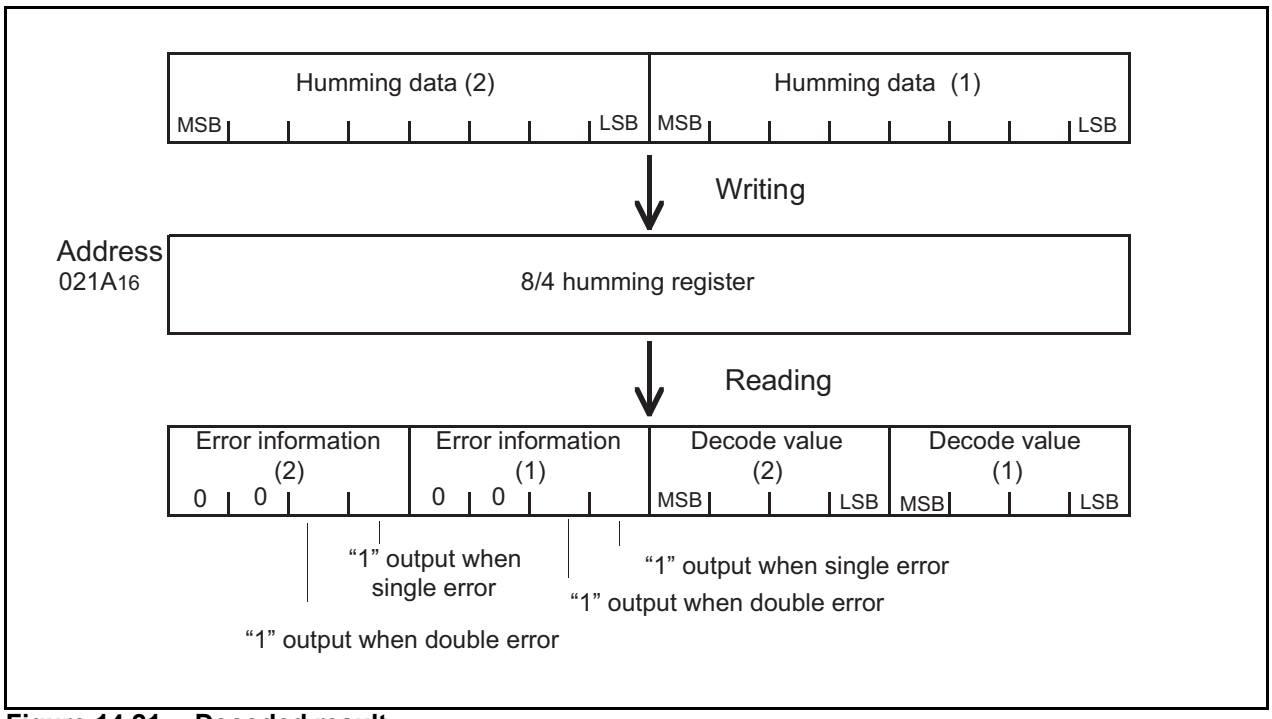



**Figure 14.21 Decoded result**

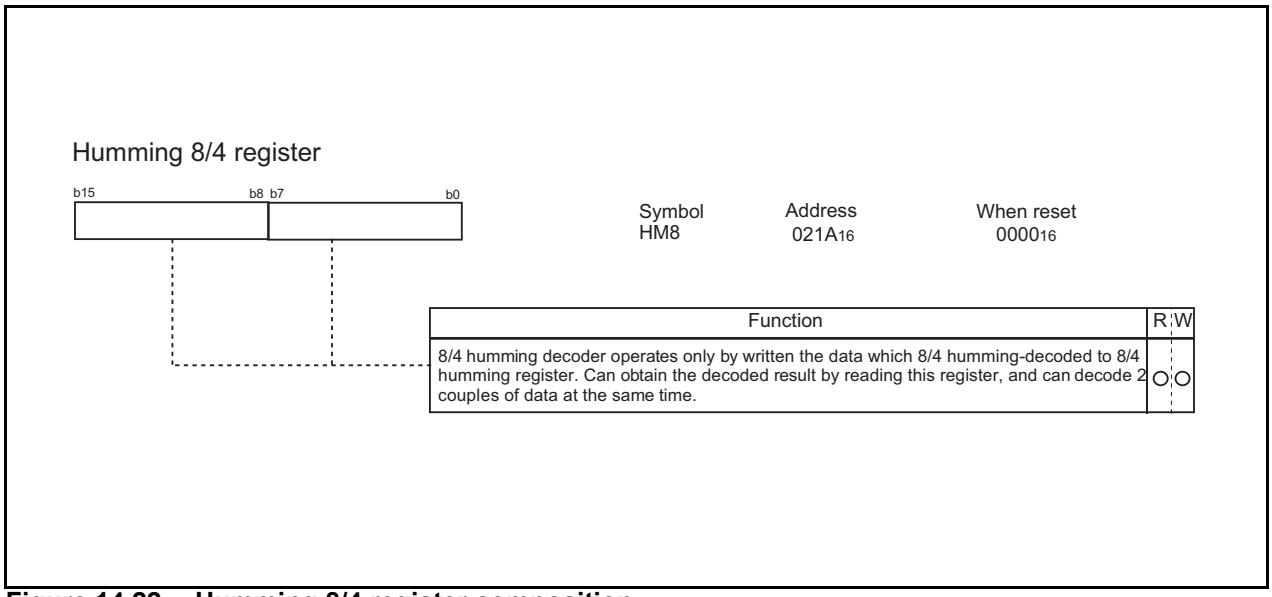



**Figure 14.22 Humming 8/4 register composition**

## 14.8 24/18 Humming Decoder

24/18 humming decoder operates only by written the data which is 24/18 humming-encoded to 24/18 humming register 0 (address 021C16) and 1 (address 021E16). Can obtain the decoded result by reading the same 24/18 humming register, and the decoded value and error information are output.
Decoded result is shown in Figure 14.23 and humming 24/18 register composition is shown in Figure 14.24.

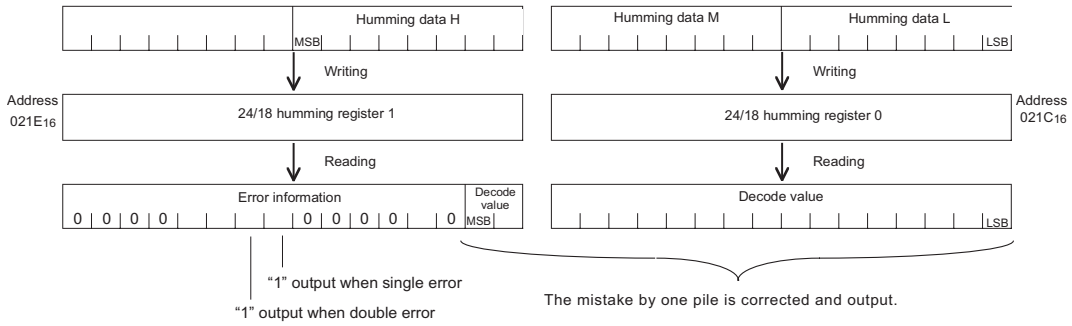



**Figure 14.23 Decoded result**

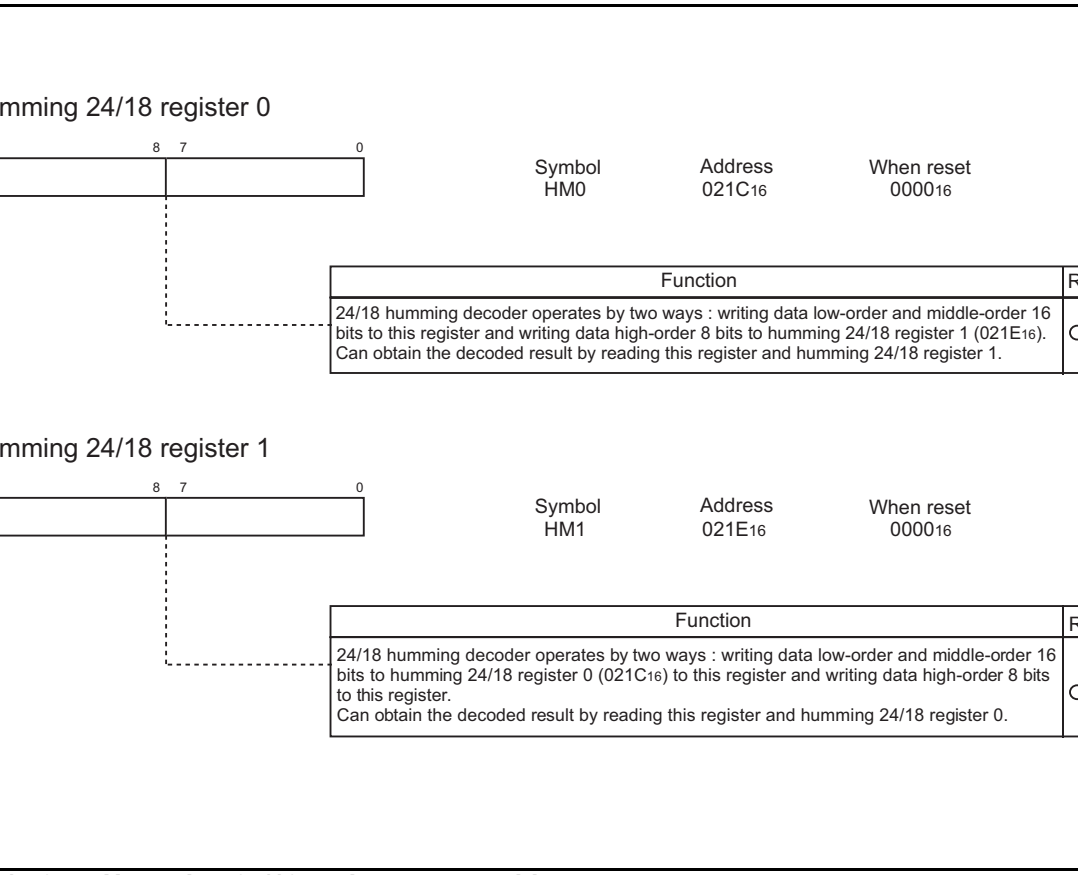



**Figure 14.24 Humming 24/18 register composition**

### Continuous error correction

When uses humming 8/4 (address 021A16) at the same time as humming 24/18, can do the continuous error correction.
Continuous error correction sequence is shown in Figure 14.25.

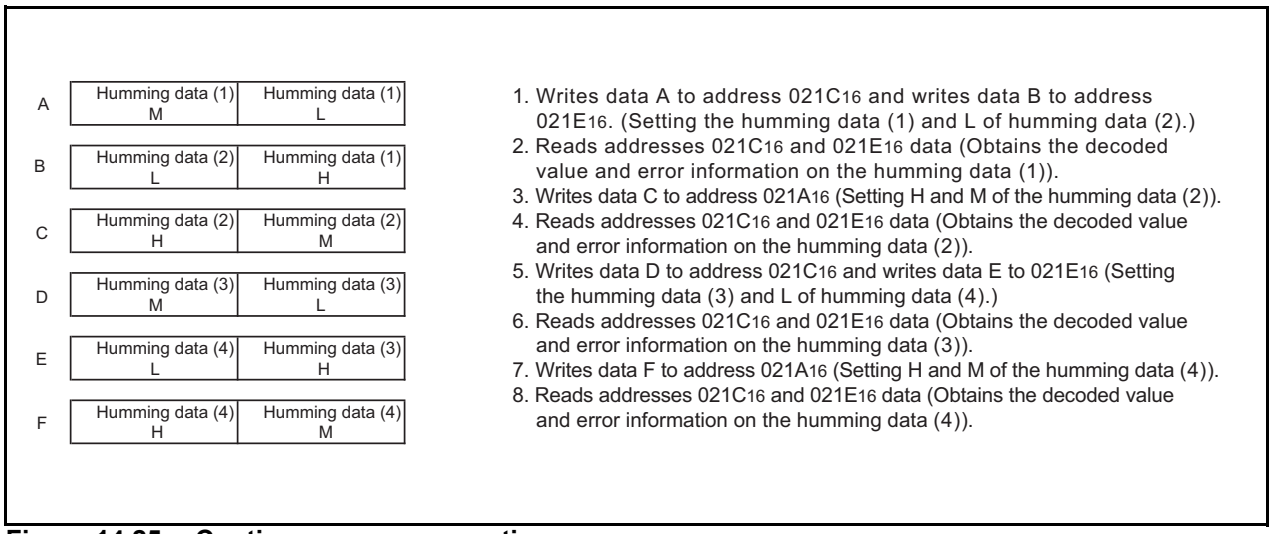
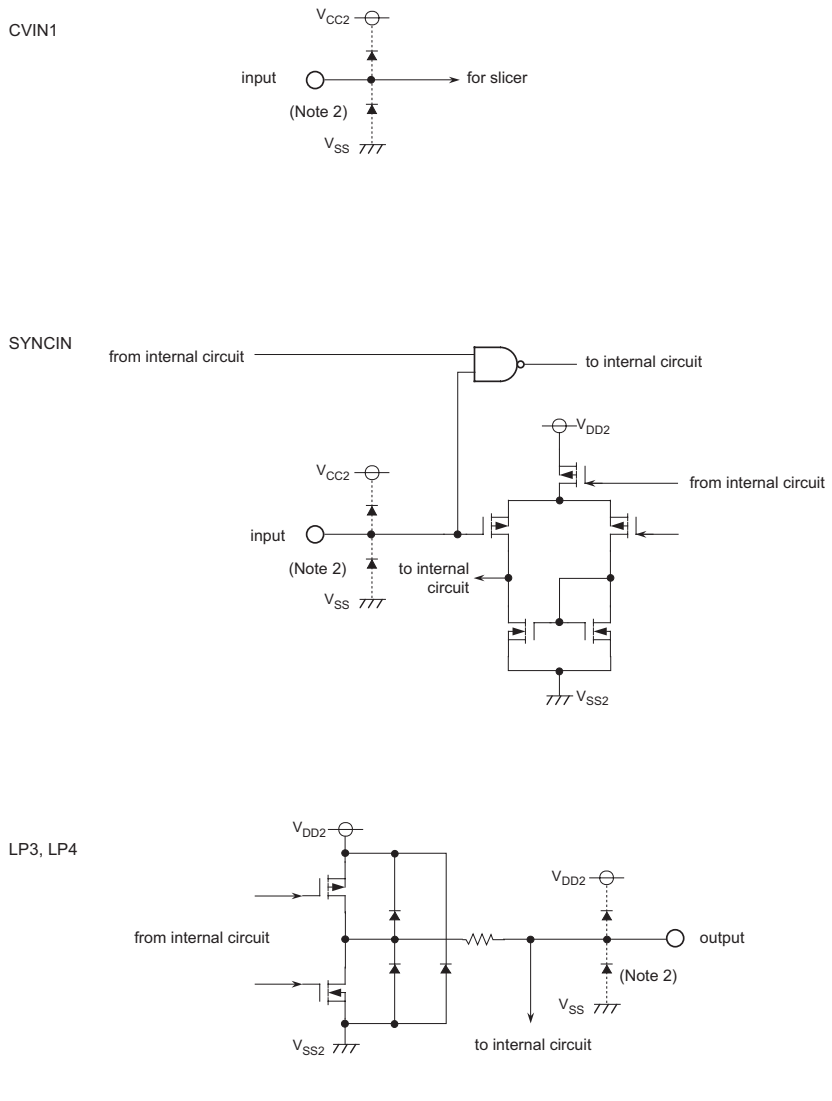



**Figure 14.25 Continuous error correction sequence**

Then, because using a part of circuit of humming 8/4 about this operation, cannot use this operation at the same time.
When using the humming circuit, do the decoded result reading operation at once after the setting data of humming. And do not access other memories (Including the humming circuit) before reading of the decoded result.

## 14.9 I/O Composition of pins for Expansion Function

Figure 14.26 and figure 14.27 show pins for expansion function.




Notes 1. Refer to expansion register composition (Address $35_{16}$.)

Notes 2. ·····|◄····· This is a parasitic diode.

The applied voltage to each port should hot exceed Vcc.
(Vcc: Vcc2 for CVIN1 and SYNCIN, and VDD2 for LP3 and LP4.)

**Figure 14.26 Pins for expansion function (1)**

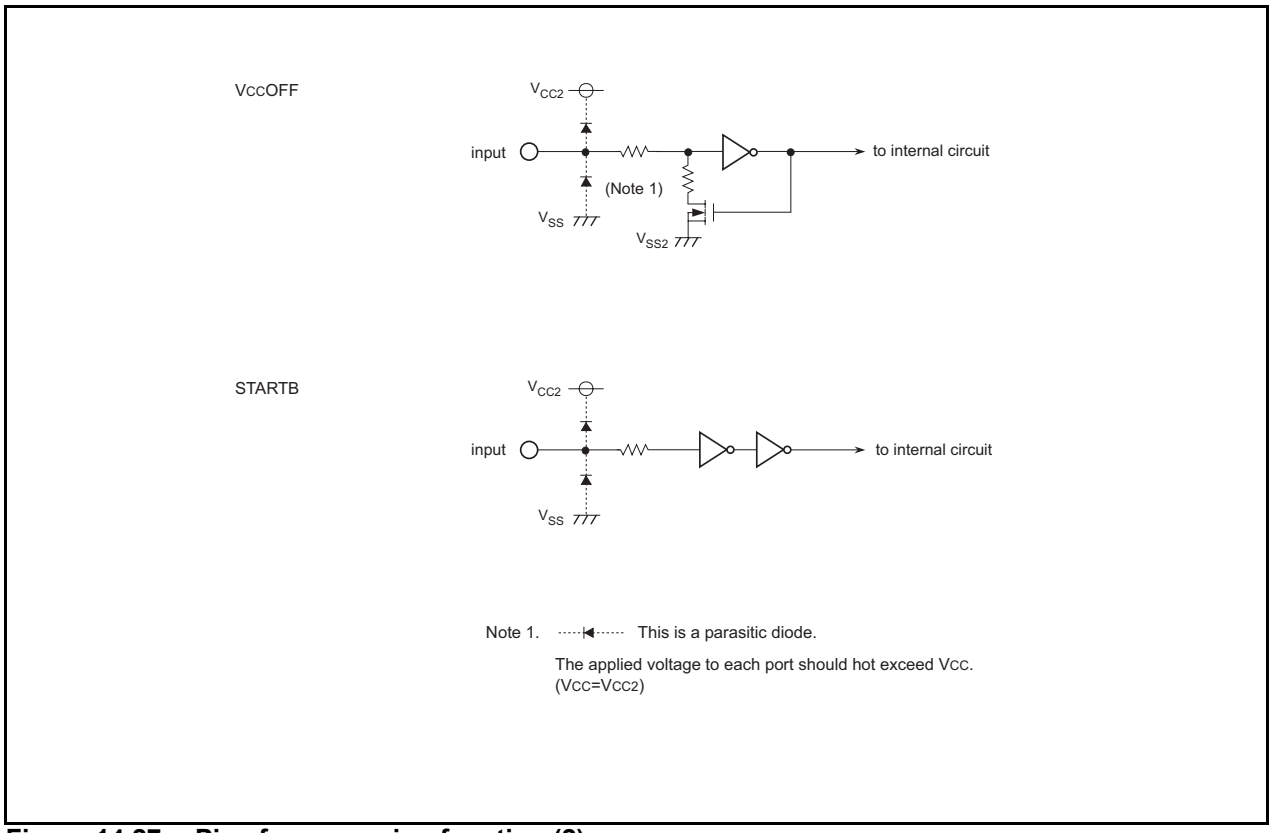



**Figure 14.27 Pins for expansion function (2)**

# 15. Programmable I/O Ports

The programmable input/output ports (hereafter referred to simply as “I/O ports”) consist of 79 lines P0 toP9 (except P85). Each port can be set for input or output every line by using a direction register, and can also be chosen to be or not be pulled high every 4 lines. P85 is an input-only port and does not have a pull-up resistor. Port P85 shares the pin with $\overline{\text{NMI}}$, so that the $\overline{\text{NMI}}$ input level can be read from the P8 register P8_5 bit.
Figures 15.1 to 15.5 show the I/O ports. Figure 15.6 shows the I/O pins.
Each pin functions as an I/O port, a peripheral function input/output.
For details on how to set peripheral functions, refer to each functional description in this manual. If any pin is used as a peripheral function input, set the direction bit for that pin to “0” (input mode). Any pin used as an output pin for peripheral functions is directed for output no matter how the corresponding direction bit is set.

## 15.1 Port Pi Direction Register (PDi Register, i = 0 to 9)

Figure 15.7 shows the direction registers.
This register selects whether the I/O port is to be used for input or output. The bits in this register correspond one for one to each port.
No direction register bit for P85 is available.

## 15.2 Port Pi Register (Pi Register, i = 0 to 9)

Figure 15.8 show the Pi registers.
Data input/output to and from external devices are accomplished by reading and writing to the Pi register. The Pi register consists of a port latch to hold the output data and a circuit to read the pin status. For ports set for input mode, the input level of the pin can be read by reading the corresponding Pi register, and data can be written to the port latch by writing to the Pi register.
For ports set for output mode, the port latch can be read by reading the corresponding Pi register, and data can be written to the port latch by writing to the Pi register. The data written to the port latch is output from the pin. The bits in the Pi register correspond one for one to each port.

## 15.3 Pull-up Control Register 0 to Pull-up Control Register 2 (PUR0 to PUR2 Registers)

Figure 15.9 shows the PUR0 to PUR2 registers.
The PUR0 to PUR2 register bits can be used to select whether or not to pull the corresponding port high in 4 bit units. The port chosen to be pulled high has a pull-up resistor connected to it when the direction bit is set for input mode.

## 15.4 Port Control Register

Figure 15.10 shows the port control register.
When the P1 register is read after setting the PCR register’s PCR0 bit to “1”, the corresponding port latch can be read no matter how the PD1 register is set.

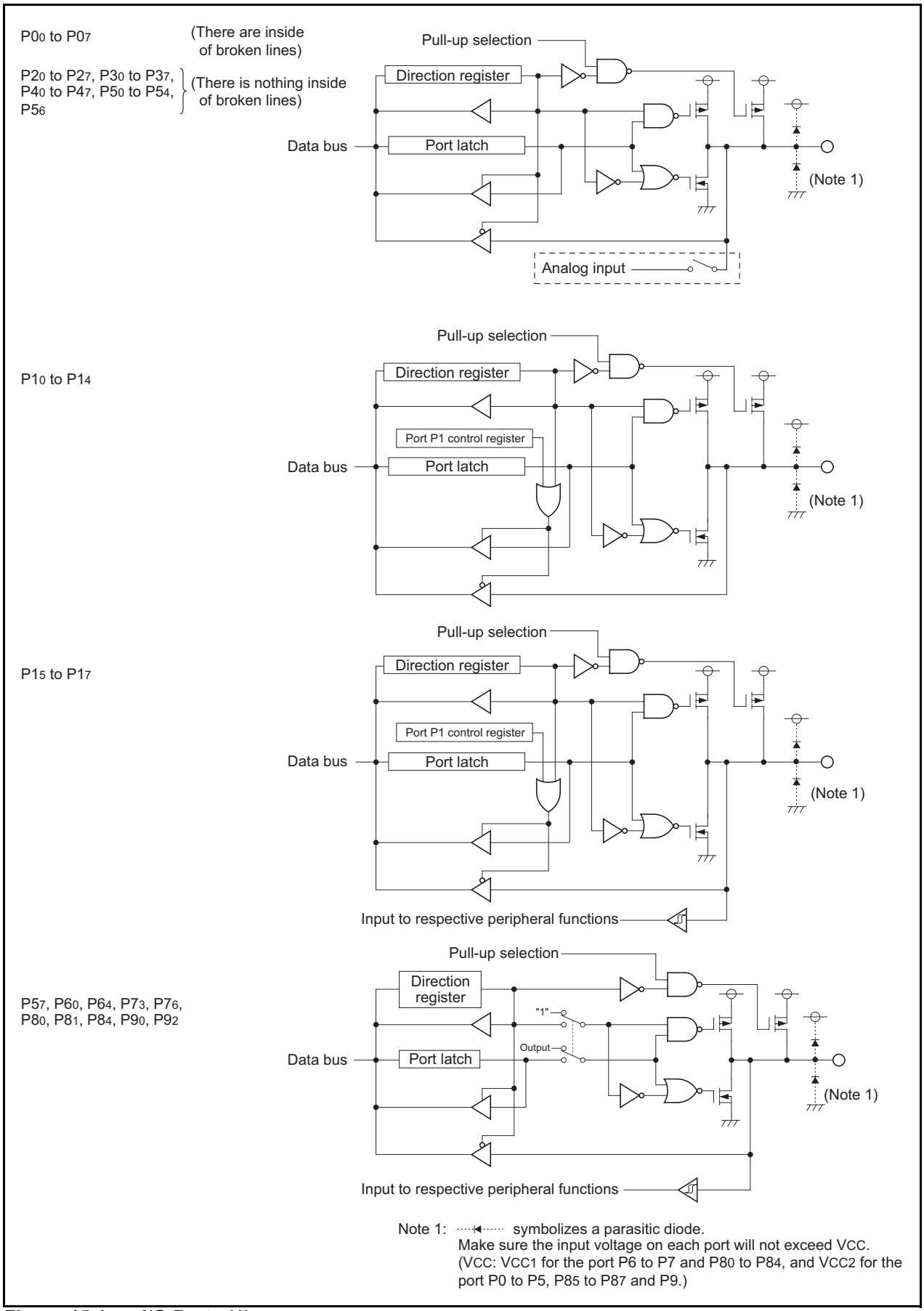
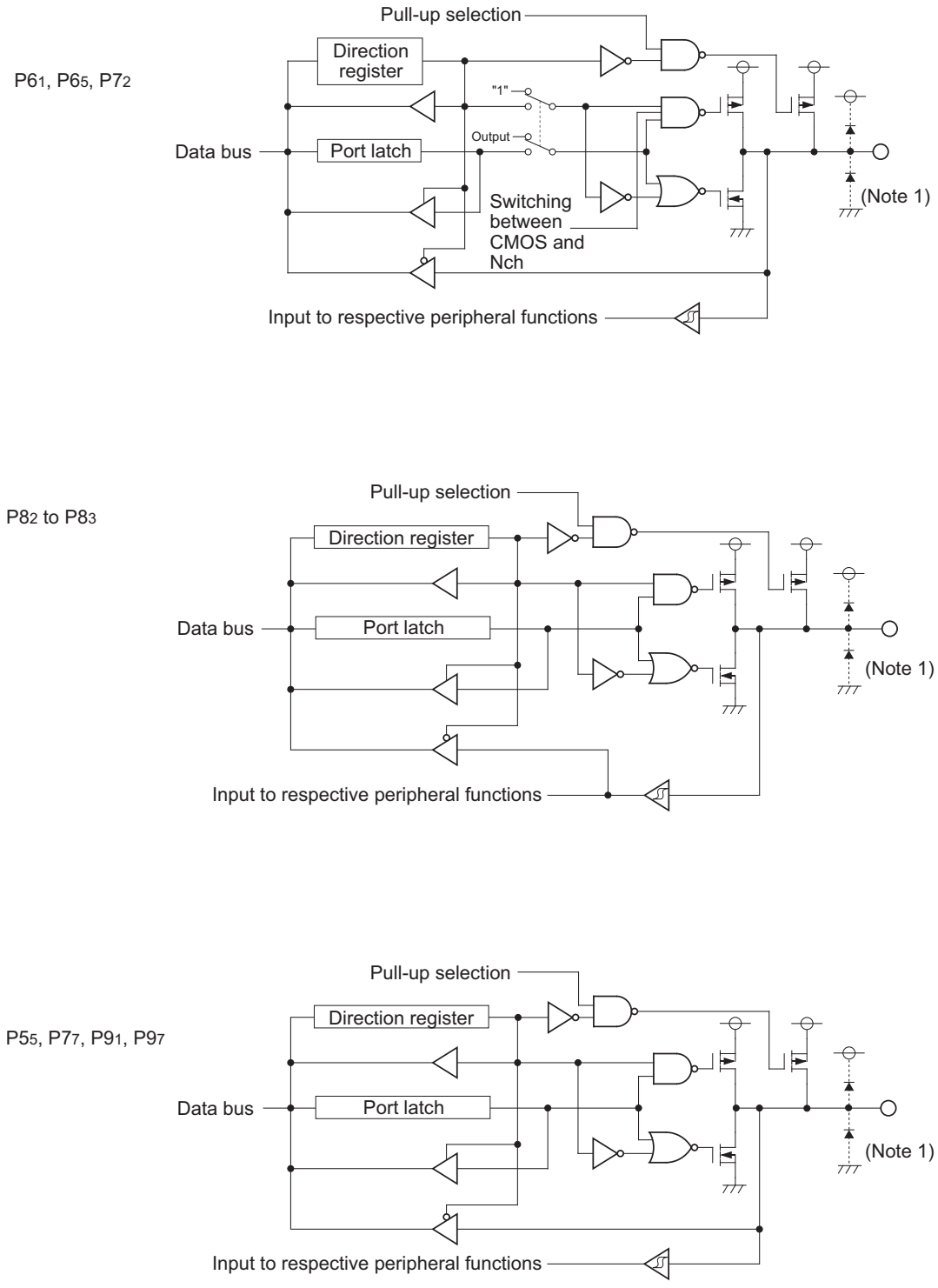



**Figure 15.1 I/O Ports (1)**




Note 1: ⋯◂⋯ symbolizes a parasitic diode.
Make sure the input voltage on each port will not exceed VCC.
(VCC: VCC1 for the port P6 to P7 and P80 to P84, and VCC2 for the port P0 to P5, P85 to P87 and P9.)

**Figure 15.2 I/O Ports (2)**

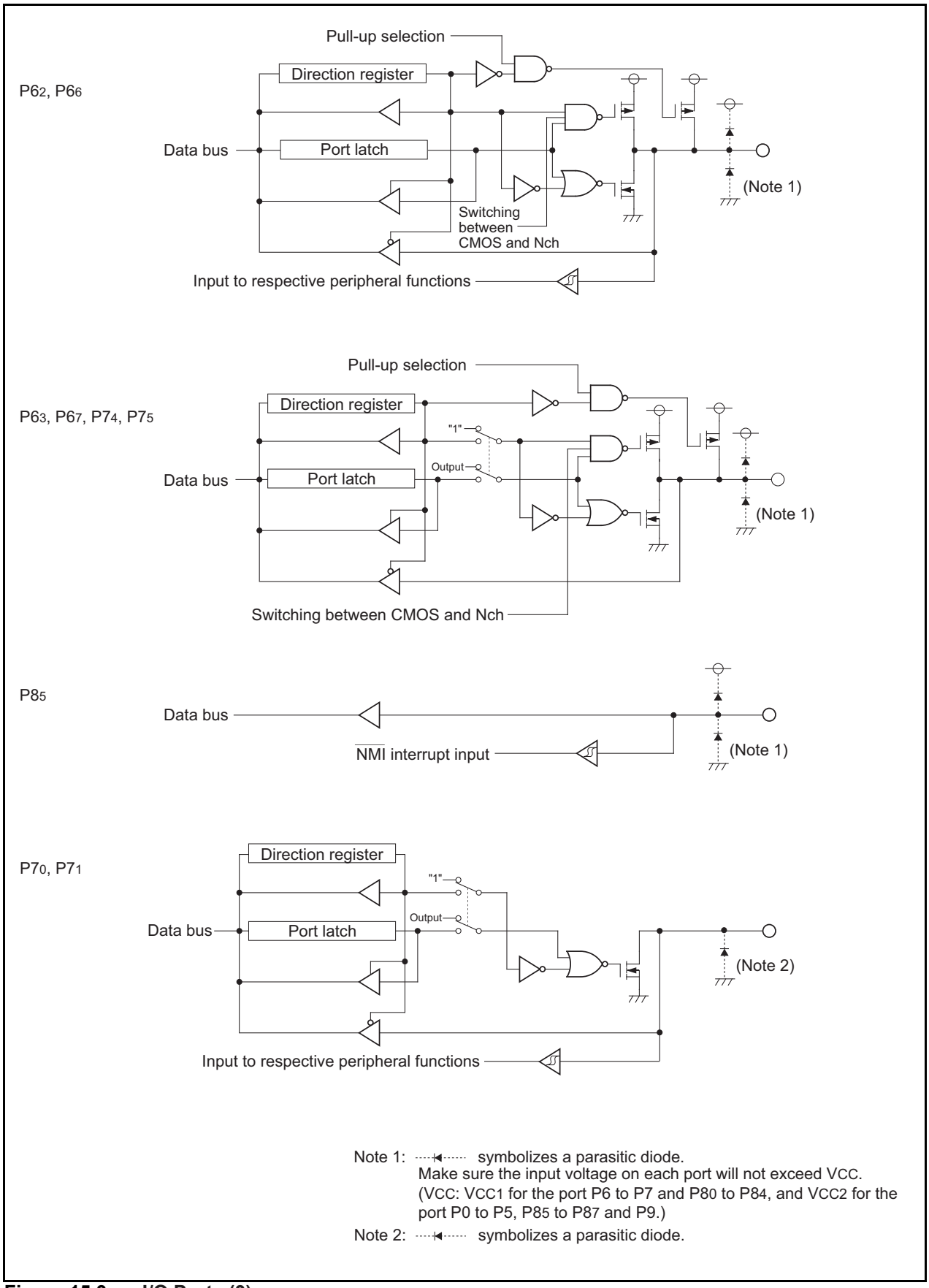



**Figure 15.3 I/O Ports (3)**

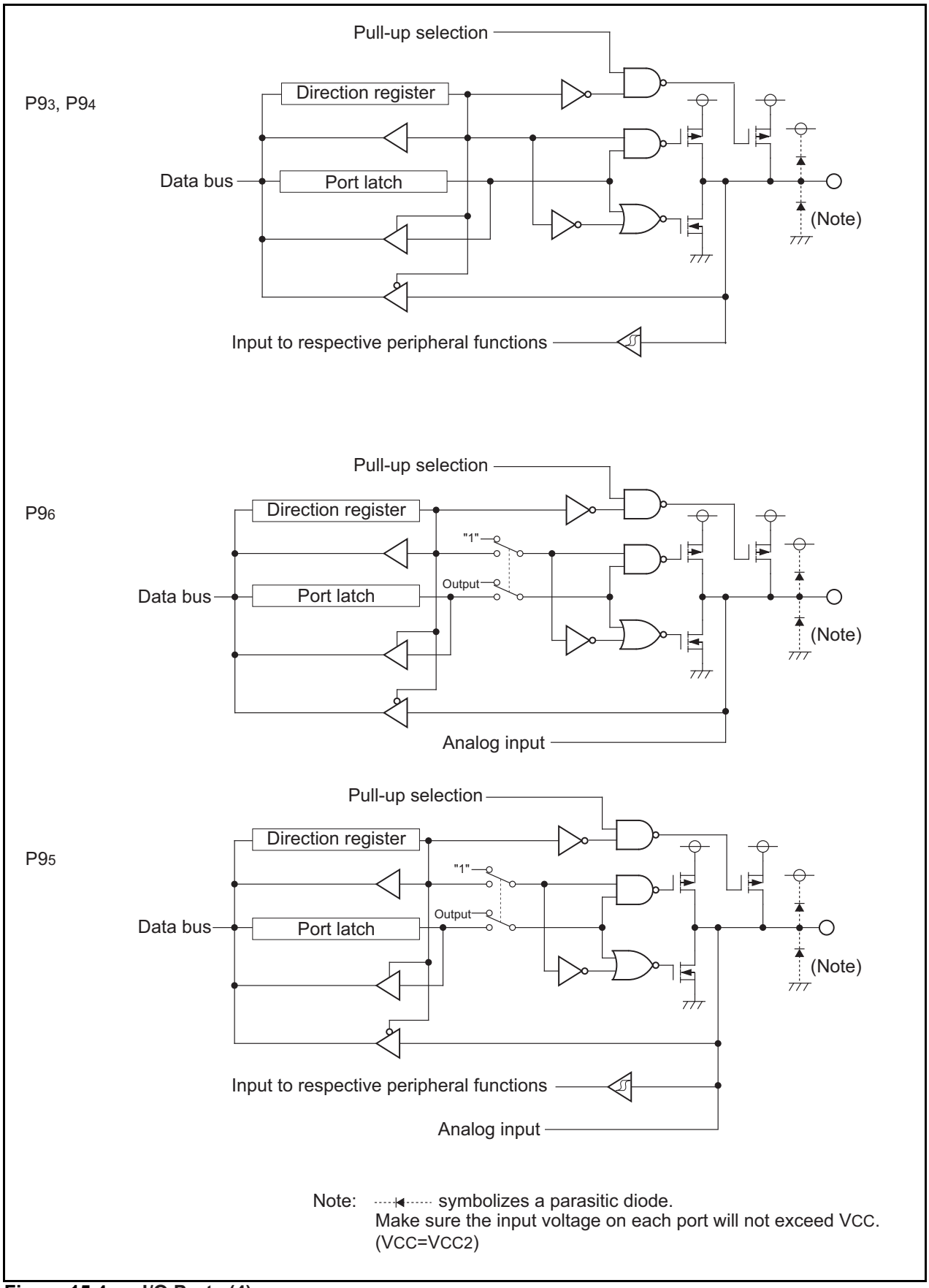



**Figure 15.4 I/O Ports (4)**

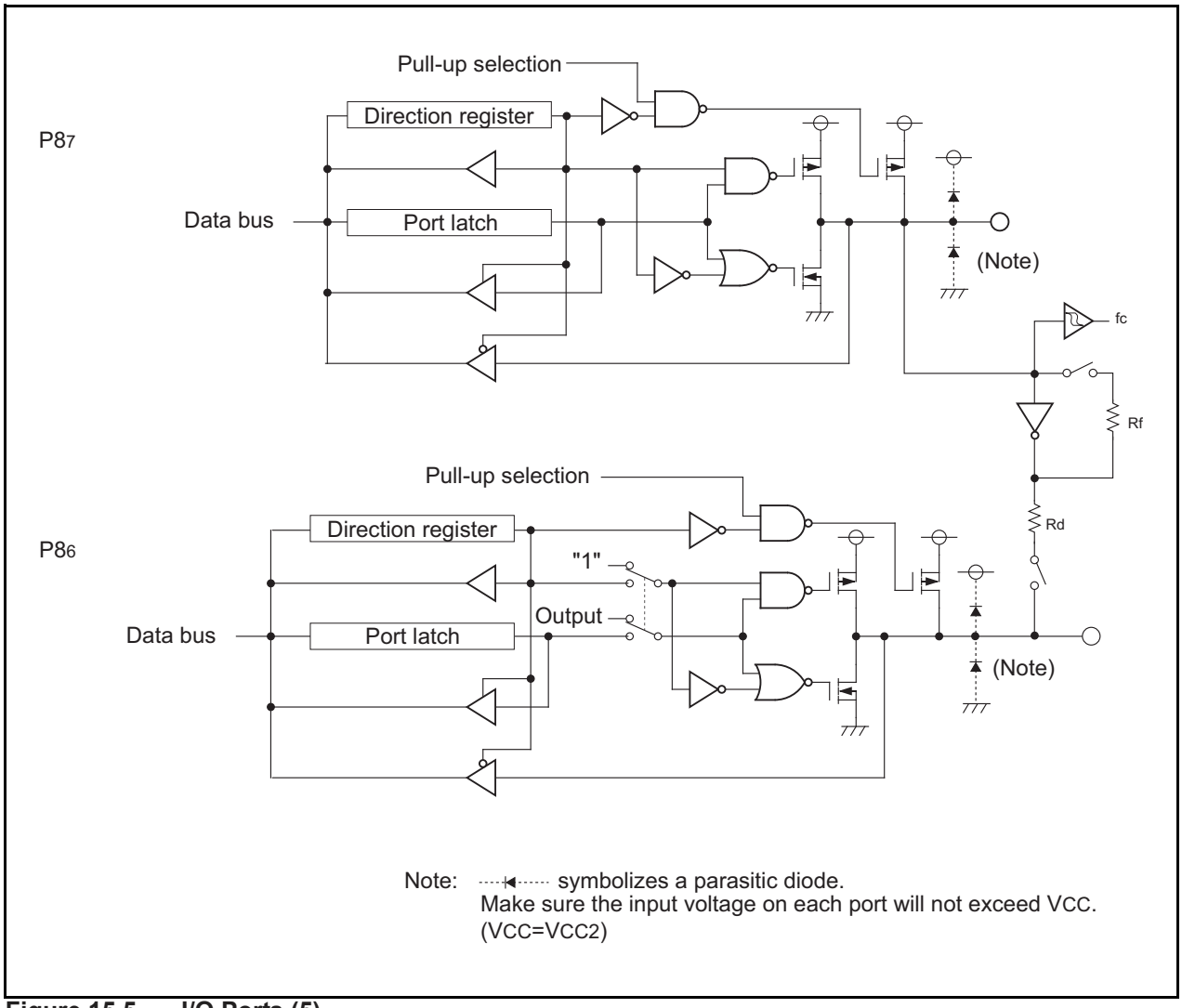



**Figure 15.5 I/O Ports (5)**

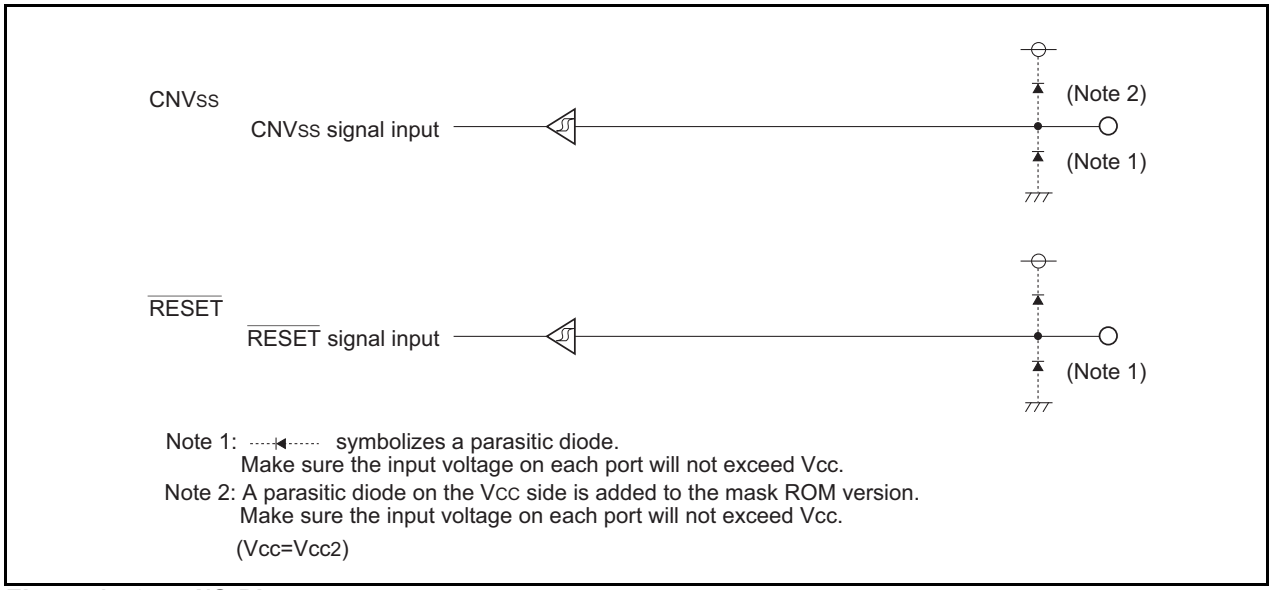
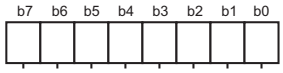



**Figure 15.6 I/O Pins**

Port Pi direction register (i=0 to 7 and 9) (Note 1, 2)

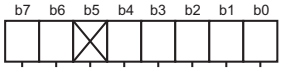



| Symbol | Address | After reset |
|---|---|---|
| PD0 to PD3 | 03E2$_{16}$, 03E3$_{16}$, 03E6$_{16}$, 03E7$_{16}$ | 00$_{16}$ |
| PD4 to PD7 | 03EA$_{16}$, 03EB$_{16}$, 03EE$_{16}$, 03EF$_{16}$ | 00$_{16}$ |
| PD9 | 03F3$_{16}$ | 00$_{16}$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| PDi_0 | Port Pi$_0$ direction bit | 0 : Input mode (Functions as an input port)<br>1 : Output mode (Functions as an output port)<br>(i = 0 to 7 and 9) | RW |
| PDi_1 | Port Pi$_1$ direction bit | | RW |
| PDi_2 | Port Pi$_2$ direction bit | | RW |
| PDi_3 | Port Pi$_3$ direction bit | | RW |
| PDi_4 | Port Pi$_4$ direction bit | | RW |
| PDi_5 | Port Pi$_5$ direction bit | | RW |
| PDi_6 | Port Pi$_6$ direction bit | | RW |
| PDi_7 | Port Pi$_7$ direction bit | | RW |

Note 1: Make sure the PD9 register is written to by the next instruction after setting the PRCR register's PRC2 bit to "1" (write enabled).

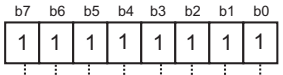
Port P8 direction register




| Symbol | Address | After reset |
|---|---|---|
| PD8 | 03F2$_{16}$ | 00X00000$_2$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| PD8_0 | Port P8$_0$ direction bit | 0 : Input mode (Functions as an input port)<br>1 : Output mode (Functions as an output port) | RW |
| PD8_1 | Port P8$_1$ direction bit | | RW |
| PD8_2 | Port P8$_2$ direction bit | | RW |
| PD8_3 | Port P8$_3$ direction bit | | RW |
| PD8_4 | Port P8$_4$ direction bit | | RW |
| —— (b5) | Nothing is assigned. In an attempt to write to this bit, write "0". The value, if read, turns out to be indeterminate. | | — |
| PD8_6 | Port P8$_6$ direction bit | 0 : Input mode (Functions as an input port)<br>1 : Output mode (Functions as an output port) | RW |
| PD8_7 | Port P8$_7$ direction bit | | RW |

Reserved register

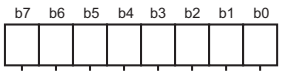



| Symbol | Address | After reset |
|---|---|---|
| RSV03F6 | 03F6$_{16}$ | 00$_{16}$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| —— (b7-b0) | Reserved bits | Must set to "1." | RW |

**Figure 15.7 PD0 to PD9 Registers**

Port Pi register (i=0 to 7 and 9) (Note 1)

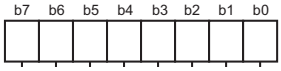



| Symbol | Address | After reset |
|---|---|---|
| P0 to P3 | $03E0_{16}$, $03E1_{16}$, $03E4_{16}$, $03E5_{16}$ | Indeterminate |
| P4 to P7 | $03E8_{16}$, $03E9_{16}$, $03EC_{16}$, $03ED_{16}$ | Indeterminate |
| P9 | $03F1_{16}$ | Indeterminate |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| Pi_0 | Port $Pi_0$ bit | The pin level on any I/O port which is set for input mode can be read by reading the corresponding bit in this register.<br>The pin level on any I/O port which is set for output mode can be controlled by writing to the corresponding bit in this register<br>0 : "L" level<br>1 : "H" level (Note 1)<br>(i = 0 to 7 and 9) | RW |
| Pi_1 | Port $Pi_1$ bit | | RW |
| Pi_2 | Port $Pi_2$ bit | | RW |
| Pi_3 | Port $Pi_3$ bit | | RW |
| Pi_4 | Port $Pi_4$ bit | | RW |
| Pi_5 | Port $Pi_5$ bit | | RW |
| Pi_6 | Port $Pi_6$ bit | | RW |
| Pi_7 | Port $Pi_7$ bit | | RW |

Note 1: Since $P7_0$ and $P7_1$ are N-channel open drain ports, the data is high-impedance.

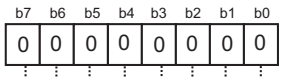
Port P8 register




| Symbol | Address | After reset |
|---|---|---|
| P8 | $03F0_{16}$ | Indeterminate |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| P8_0 | Port $P8_0$ bit | The pin level on any I/O port which is set for input mode can be read by reading the corresponding bit in this register.<br>The pin level on any I/O port which is set for output mode can be controlled by writing to the corresponding bit in this register (except for $P8_5$)<br>0 : "L" level<br>1 : "H" level | RW |
| P8_1 | Port $P8_1$ bit | | RW |
| P8_2 | Port $P8_2$ bit | | RW |
| P8_3 | Port $P8_3$ bit | | RW |
| P8_4 | Port $P8_4$ bit | | RW |
| P8_5 | Port $P8_5$ bit | | RO |
| P8_6 | Port $P8_6$ bit | | RW |
| P8_7 | Port $P8_7$ bit | | RW |

Reserved register




| Symbol | Address | After reset |
|---|---|---|
| RSV03F4 | $03F4_{16}$ | Indeterminate |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| $\overline{\text{(b7-b0)}}$ | Reserved bits | Must set to "0." | RW |

**Figure 15.8 P0 to P9 Registers**

## Pull-up control register 0 (Note 1)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

| Symbol | Address | After reset |
|---|---|---|
| PUR0 | 03FC16 | 0016 |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| PU00 | P00 to P03 pull-up | 0 : Not pulled high<br>1 : Pulled high (Note 1) | RW |
| PU01 | P04 to P07 pull-up | | RW |
| PU02 | P10 to P13 pull-up | | RW |
| PU03 | P14 to P17 pull-up | | RW |
| PU04 | P20 to P23 pull-up | | RW |
| PU05 | P24 to P27 pull-up | | RW |
| PU06 | P30 to P33 pull-up | | RW |
| PU07 | P34 to P37 pull-up | | RW |

Note 1: The pin for which this bit is "1" (pulled high) and the direction bit is "0" (input mode) is pulled high.

## Pull-up control register 1

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

| Symbol | Address | After reset (Note 3) |
|---|---|---|
| PUR1 | 03FD16 | 000000002 |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| PU10 | P40 to P43 pull-up | 0 : Not pulled high<br>1 : Pulled high (Note 2) | RW |
| PU11 | P44 to P47 pull-up | | RW |
| PU12 | P50 to P53 pull-up | | RW |
| PU13 | P54 to P57 pull-up | | RW |
| PU14 | P60 to P63 pull-up | | RW |
| PU15 | P64 to P67 pull-up | | RW |
| PU16 | P72 to P73 pull-up (Note 1) | | RW |
| PU17 | P74 to P77 pull-up | | RW |

Note 1: The P70 and P71 pins do not have pull-ups.
Note 2: The pin for which this bit is "1" (pulled high) and the direction bit is "0" (input mode) is pulled high.
Note 3: The values after hardware reset are as follows:
- 000000002 when input on CNVss pin is "L"

The values after software reset and watchdog timer reset are as follows:
- 000000002 when PM 01 to PM00 bits of PM0 register are "002" (single-chip mode)

## Pull-up control register 2

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| X | X | 0 | 0 | | | | |

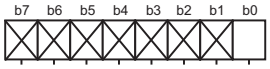
| Symbol | Address | After reset |
|---|---|---|
| PUR2 | 03FE16 | 0016 |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| PU20 | P80 to P83 pull-up | 0 : Not pulled high<br>1 : Pulled high (Note 1) | RW |
| PU21 | P84 to P87 pull-up (Note 2) | | RW |
| PU22 | P90 to P93 pull-up | | RW |
| PU23 | P94 to P97 pull-up | | RW |
| (b5-b4) | Must set to "0" | | RW |
| —— (b7-b6) | Nothing is assigned. In an attempt to write to these bits, write "0". The value, if read, turns out to be "0". | | —— |

Note 1: The pin for which this bit is "1" (pulled high) and the direction bit is "0" (input mode) is pulled high.
Note 2: The P85 pin does not have pull-up.

**Figure 15.9 PUR0 to PUR2 Registers**

Port control register




Symbpl: PCR
Address: 03FF16
After reset: 0016

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| PCR0 | Port P1 control bit | Operation performed when the P1 register is read<br>0: When the port is set for input, the input levels of P10 to P17 pins are read. When set for output, the port latch is read.<br>1: The port latch is read regardless of whether the port is set for input or output. | RW |
| —<br>(b7-b1) | Nothing is assigned. In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”. | | — |

**Figure 15.10 PCR Register**

**Table 15.1 Unassigned Pin Handling in Single-chip Mode**

| Pin name | Connection |
|---|---|
| Ports P0 to P7, P80 to P84, P86 to P87, P9 | After setting for input mode, connect every pin to VSS via a resistor(pull-down); or after setting for output mode, leave these pins open. (Note 1, 2 ,3) |
| XOUT (Note 4) | Open |
| $\overline{\text{NMI}}$ (P85) | Connect via resistor to VCC (pull-up) |
| AVCC | Connect to VCC |
| AVSS | Connect to VSS |

Note 1: When setting the port for output mode and leave it open, be aware that the port remains in input mode until it is switched to output mode in a program after reset. For this reason, the voltage level on the pin becomes indeterminate, causing the power supply current to increase while the port remains in input mode. Furthermore, by considering a possibility that the contents of the direction registers could be changed by noise or noise-induced runaway, it is recommended that the contents of the direction registers be periodically reset in software, for the increased reliability of the program.
Note 2: Make sure the unused pins are processed with the shortest possible wiring from the microcomputer pins (within 2 cm).
Note 3: When the ports P70 and P71 are set for output mode, make sure a low-level signal is output from the pins. The ports P70 and P71 are N-channel open-drain outputs.
Note 4: With external clock input to XIN pin.

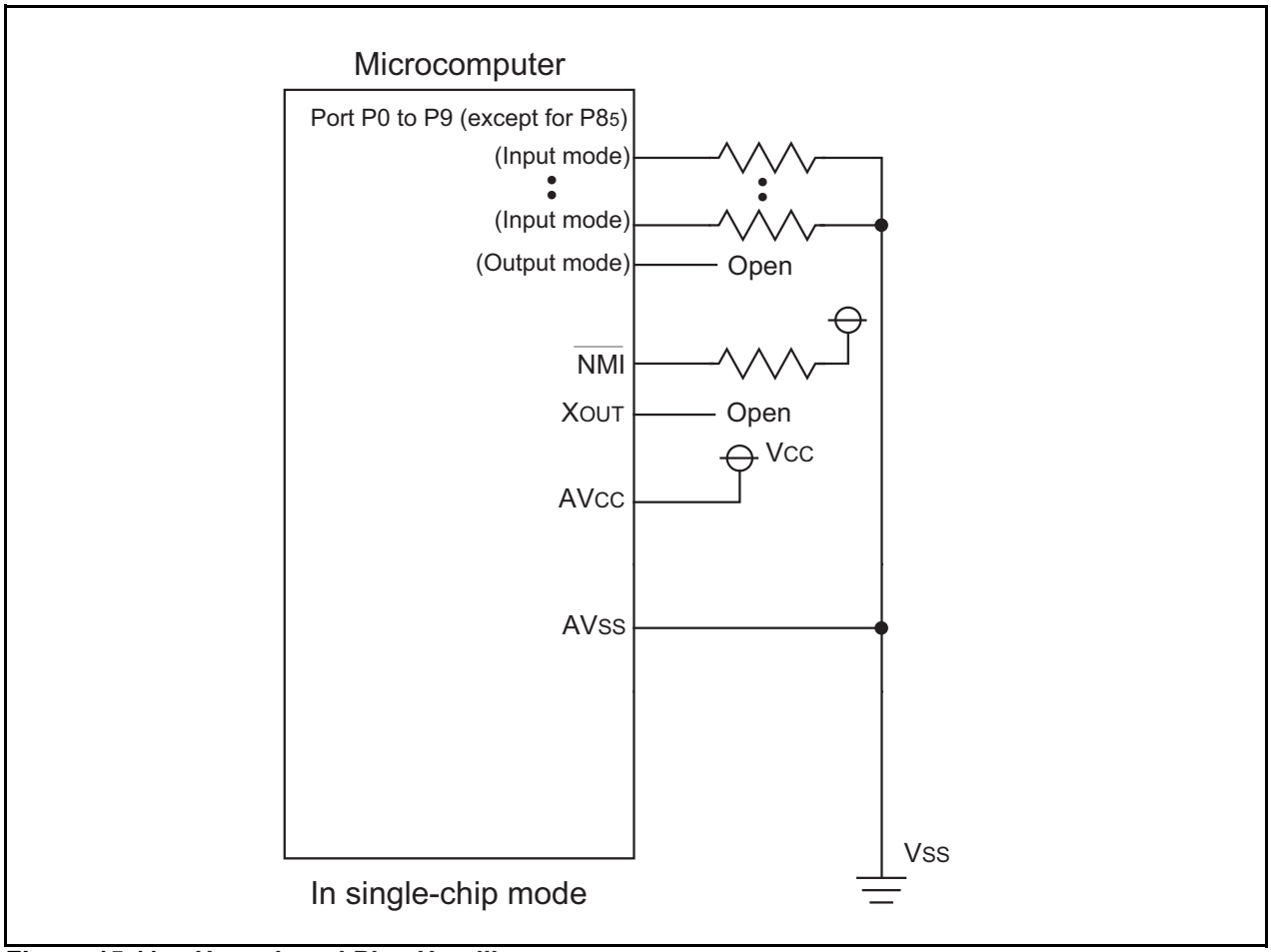



**Figure 15.11 Unassigned Pins Handling**

# 16. Electrical Characteristics

**Table 16.1 Absolute Maximum Ratings**

| Symbol | Parameter | | Condition | Rated value | Unit |
|---|---|---|---|---|---|
| $V_{CC1}$, $V_{CC2}$ | Supply voltage | | $V_{CC2}$=AVcc | -0.3 to 6.0 | V |
| $V_{CC1}$ | Supply voltage | | $V_{CC1}$ | -0.3 to $V_{CC2}$ | V |
| AVcc | Analog supply voltage | | $V_{CC2}$=AVcc | -0.3 to 6.0 | V |
| $V_{DD2}$ | Analog supply voltage | | $V_{CC2}$=$V_{DD2}$ | -0.3 to 6.0 | V |
| $V_I$ | Input voltage | $\overline{RESET}$, $CNV_{SS}$<br>$P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$,<br>$P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$,<br>$P8_5$ to $P8_7$,<br>$P9_0$ to $P9_7$,<br>$X_{IN}$, M1, STARTB | | -0.3 to $V_{CC2}$ + 0.3 | V |
| | | $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_4$ | | -0.3 to $V_{CC1}$ + 0.3 | V |
| | | $P7_0$, $P7_1$ | | -0.3 to 6.0 | V |
| $V_O$ | Output voltage | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$,<br>$P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$,<br>$P8_6$, $P8_7$, $P9_0$ to $P9_7$,<br>$X_{OUT}$ | | -0.3 to $V_{CC2}$ + 0.3 | V |
| | | $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_4$ | | -0.3 to $V_{CC1}$ + 0.3 | V |
| | | $P7_0$, $P7_1$ | | -0.3 to 6.0 | V |
| $P_d$ | Power dissipation | | Topr=25 °C | 550 | mW |
| $T_{opr}$ | Operating ambient temperature | | | -20 to 70 | °C |
| $T_{stg}$ | Storage temperature | | | -20 to 125 | °C |

Note: Following setting is required: $V_{CC1} \leq V_{CC2}$

## Table 16.2 Recommended Operating Conditions (Note 1)

| Symbol | Parameter | | Standard | | | Unit |
|---|---|---|---|---|---|---|
| | | | Min. | Typ. | Max. | |
| $V_{CC1}$, $V_{CC2}$ | Supply voltage ($V_{CC1} \leq V_{CC2}$) | | 2.0 | 5.0 | 5.5 | V |
| AVcc | Analog supply voltage | | | $V_{CC2}$ | | V |
| $V_{DD2}$ | Analog supply voltage | | | $V_{CC2}$ | | V |
| Vss | Supply voltage | | | 0 | | V |
| AVss | Analog supply voltage | | | 0 | | V |
| $V_{IH}$ | HIGH input voltage | $P3_1$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$ | 0.8$V_{CC2}$ | | $V_{CC2}$ | V |
| | | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ | 0.8$V_{CC2}$ | | $V_{CC2}$ | V |
| | | $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_4$ | 0.8$V_{CC1}$ | | $V_{CC1}$ | V |
| | | $P8_5$ to $P8_7$, $P9_0$ to $P9_7$<br>$X_{IN}$, $\overline{RESET}$, $CNV_{SS}$, M1, STARTB | 0.8$V_{CC2}$ | | $V_{CC2}$ | V |
| | | $P7_0$, $P7_1$ | 0.8$V_{CC1}$ | | 5.75 | V |
| $V_{IL}$ | LOW input voltage | $P3_1$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$ | 0 | | 0.2$V_{CC2}$ | V |
| | | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ | 0 | | 0.2$V_{CC2}$ | V |
| | | $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_4$ | 0 | | 0.2$V_{CC1}$ | V |
| | | $P8_5$ to $P8_7$, $P9_0$ to $P9_7$,<br>$X_{IN}$, $\overline{RESET}$, $CNV_{SS}$, M1, STARTB | 0 | | 0.2$V_{CC2}$ | V |
| $V_{CVIN}$ | Composite video input voltage CVIN, SYNCIN | | | 2$V_{P-P}$ | | V |
| $I_{OH\ (peak)}$ | HIGH peak output current (Note2, Note3) | $P0_0$ to $P0_7$,$P1_0$ to $P1_7$, $P2_0$ to $P2_7$,$P3_0$ to $P3_7$, $P4_0$ to $P4_7$,$P5_0$ to $P5_7$, $P6_0$ to $P6_7$,$P7_2$ to $P7_7$, $P8_0$ to $P8_4$,$P8_6$,$P8_7$,$P9_0$ to $P9_7$ | | | - 10.0 | mA |
| $I_{OH\ (avg)}$ | HIGH average output current | $P0_0$ to $P0_7$,$P1_0$ to $P1_7$, $P2_0$ to $P2_7$,$P3_0$ to $P3_7$, $P4_0$ to $P4_7$,$P5_0$ to $P5_7$, $P6_0$ to $P6_7$,$P7_2$ to $P7_7$, $P8_0$ to $P8_4$,$P8_6$,$P8_7$,$P9_0$ to $P9_7$ | | | - 5.0 | mA |
| $I_{OL\ (peak)}$ | LOW peak output current | $P0_0$ to $P0_7$,$P1_0$ to $P1_7$, $P2_0$ to $P2_7$,$P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$,$P7_0$ to $P7_7$, $P8_0$ to $P8_4$,$P8_6$,$P8_7$,$P9_0$ to $P9_7$ | | | 10.0 | mA |
| $I_{OL\ (avg)}$ | LOW average output current | $P0_0$ to $P0_7$,$P1_0$ to $P1_7$, $P2_0$ to $P2_7$,$P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$,$P7_0$ to $P7_7$, $P8_0$ to $P8_4$,$P8_6$,$P8_7$,$P9_0$ to $P9_7$ | | | 5.0 | mA |
| f ($X_{IN}$) | Main clock input oscillation frequency (Note 4) | $V_{CC2}$=2.9 to 5.5V | 0 | | 16 | MHz |
| f ($X_{CIN}$) | Sub-clock oscillation frequency | $V_{CC2}$=2.0 to 5.5V(Note 5) | | 32.768 | 50 | kHz |
| f (BCLK) | CPU operation clock | | 0 | | 16 | MHz |

Note 1: Referenced to $V_{CC}$ = $V_{CC1}$ = $V_{CC2}$ = 2.0 to 5.5V at Topr = -20 to 70 °C unless otherwise specified.

Note 2: The mean output current is the mean value within 100ms.

Note 3: The total $I_{OL}$ (peak) for ports P0, P1, P2, P3, P4, P5,$P8_6$, $P8_7$, P9 must be 80mA max. The total $I_{OL}$ (peak) for ports P6, P7and$P8_0$ to $P8_4$ must be 80mA max. The total $I_{OH}$ (peak) for ports P0, P1, and P2 must be -40mA max. The total $I_{OH}$ (peak) for ports P3, P4andP5 must be -40mA max.

Note 4: Use the $V_{CC1}$ and $V_{CC2}$ power supply voltage on the following conditions.
- $V_{CC1}$ = 3.00V to $V_{CC2}$, $V_{CC2}$ = 4.00V to 5.5V (at f($X_{IN}$) = 16MHz)
- $V_{CC1}$ = 2.90V to $V_{CC2}$, $V_{CC2}$ = 2.90V to 5.5V (at f($X_{IN}$) = 16MHz, at divide-by-8 or 16)

Note 5: Use in low power dissipation mode. When operating on low voltage ($V_{CC}$ = 3.0V), only single-chip mode can be used. If the $V_{CC2}$ supply voltage is less than 2.6 V, be aware that only the CPU, RAM, clock timer, interrupt, and Input/Output ports can be used. Other control circuits (e.g., timers A and B, serial I/O, UART) cannot be used.

**Table 16.3 A/D Conversion Characteristics (Note 1)**

| Symbol | Parameter | Measuring condition | | Standard | | | Unit |
|---|---|---|---|---|---|---|---|
| | | | | Min. | Typ. | Max. | |
| – | Resolution | $V_{REF}$ =$V_{CC}$ | | | | 8 | Bits |
| – | Absolute accuracy | $V_{REF}$= $V_{CC}$ = 5V | $AN_0$ to $AN_7$ input | | | ±3 | LSB |
| | | | ANEX0, ANEX1 input External operation amp | | | ±4 | LSB |
| $t_{CONV}$ | Conversion time(8bit), Sample & hold function available | $V_{REF}$ =$V_{CC}$=5V, ØAD=10MHz | | 2.8 | | | µs |
| $t_{SAMP}$ | Sampling time | | | 0.3 | | | µs |
| $V_{REF}$ | Reference voltage | | | 4.5 | | $V_{CC}$ | V |
| $V_{IA}$ | Analog input voltage | | | 0 | | $V_{REF}$ | V |

Note 1: Referenced to $V_{CC2}$=$AV_{CC}$=$V_{REF}$=4.5 to 5.5V, $V_{SS}$=$AV_{SS}$=0V at Topr = -20 to 70 ˚C unless otherwise specified.
Note 2: AD operation clock frequency (ØAD frequency) must be 10 MHz or less.
Note 3: A case without sample & hold function turn ØAD frequency into 250 kHz or more.
A case with sample & hold function turn ØAD frequency into 1 MHz or more.

**Table 16.4 Flash Memory Version Electrical Characteristics (Note 1)**

| Symbol | Parameter | Measuring condition | Standard | | | Unit |
|---|---|---|---|---|---|---|
| | | | Min. | Typ. | Max | |
| — | Word program time | | | 30 | 200 | µs |
| — | Block erase time | | | 1 | 4 | s |
| — | Lock bit program time | | | 30 | 200 | µs |
| tps | Flash memory circuit stabilization wait time | | | | 15 | µs |

Note 1: Referenced to $V_{CC2}$=4.75 to 5.25V at Topr = 0 to 60 ˚C unless otherwise specified.

**Table 16.5 Flash Memory Version Program/Erase Voltage and Read Operation Voltage Characteristics (Topr = 0 to 60°C**

| Flash program, erase voltage | Flash read operation voltage |
|---|---|
| $V_{CC2}$ = 5.0 ± 0.25 V | $V_{CC2}$ = 2.0 to 5.5 V |

**Table 16.6 Power Supply Circuit Timing Characteristics**

| Symbol | Parameter | Measuring condition | Standard | | | Unit |
|---|---|---|---|---|---|---|
| | | | Min. | Typ. | ax. | |
| td(P-R) | ime for internal power supply stabilization during powering-on | VCC=5.0V | | | 2 | ms |
| td(R-S) | STOP release time | | | | 150 | µs |
| td(W-S) | Low power dissipation mode wait mode release time | | | | 150 | µs |

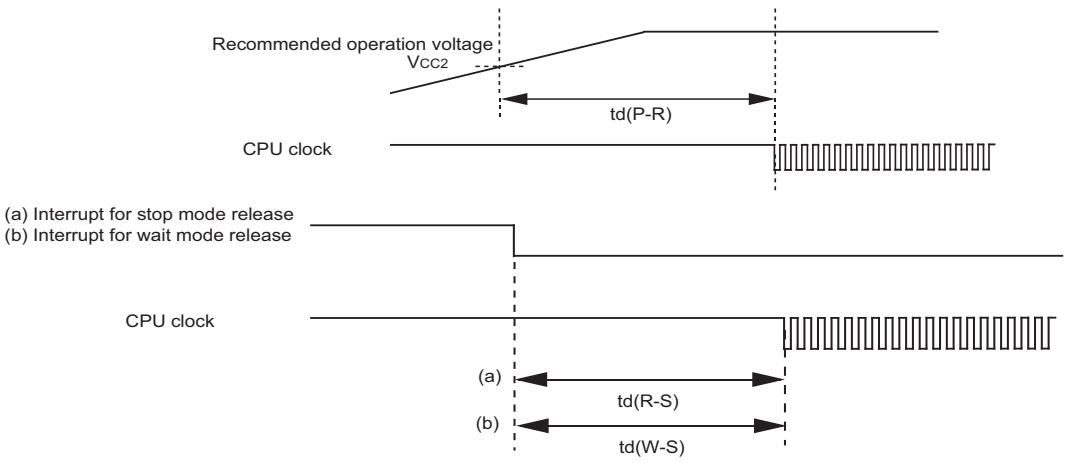


**Table 16.7 Electrical Characteristics (1) (Note 1)**

$V_{CC1} = V_{CC2} = 5V$

| Symbol | Parameter | | | Measuring condition | Standard Min | Standard Typ. | Standard Max. | Unit |
|---|---|---|---|---|---|---|---|---|
| $V_{OH}$ | HIGH output voltage | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$ | | $I_{OH}$=-5mA | $V_{CC2}$-2.0 | | $V_{CC2}$ | V |
| | | $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_4$ | | $I_{OH}$=-5mA | $V_{CC1}$-2.0 | | $V_{CC1}$ | V |
| $V_{OH}$ | HIGH output voltage | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$ | | $I_{OH}$=-200µA | $V_{CC2}$-0.3 | | $V_{CC2}$ | V |
| | | $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_4$ | | $I_{OH}$=-200µA | $V_{CC1}$-0.3 | | $V_{CC1}$ | V |
| $V_{OH}$ | HIGH output voltage | LP3 , LP4 | | $V_{CC}$=4.5V, $I_{OH}$=-0.05mA | 3.75 | | | V |
| $V_{OH}$ | HIGH output voltage | $X_{OUT}$ | HIGHPOWER | $I_{OH}$=-1mA | $V_{CC2}$-2.0 | | $V_{CC2}$ | V |
| | | | LOWPOWER | $I_{OH}$=-0.5mA | $V_{CC2}$-2.0 | | $V_{CC2}$ | |
| | HIGH output voltage | $X_{COUT}$ | HIGHPOWER | With no load applied | | 2.5 | | V |
| | | | LOWPOWER | With no load applied | | 1.6 | | |
| $V_{OL}$ | LOW output voltage | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$ | | $I_{OL}$=5mA | | | 2.0 | V |
| | | $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_4$ | | $I_{OL}$=5mA | | | 2.0 | V |
| $V_{OL}$ | LOW output voltage | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$ | | $I_{OL}$=200µA | | | 0.45 | V |
| | | $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_4$ | | $I_{OL}$=200µA | | | 0.45 | V |
| $V_{OL}$ | LOW output voltage | LP3 to LP4 | | $V_{CC}$=4.5V, $I_{OL}$=0.05mA | | | 0.4 | V |
| $V_{OL}$ | LOW output voltage | $X_{OUT}$ | HIGHPOWER | $I_{OL}$=1mA | | | 2.0 | V |
| | | | LOWPOWER | $I_{OL}$=0.5mA | | | 2.0 | |
| | LOW output voltage | $X_{COUT}$ | HIGHPOWER | With no load applied | | 0 | | V |
| | | | LOWPOWER | With no load applied | | 0 | | |
| $V_{T+}$-$V_{T-}$ | Hysteresis | $TA0_{IN}$ to $TA4_{IN}$, $\overline{TB0_{IN}}$ to $\overline{TB5_{IN}}$, $\overline{INT0}$ to $\overline{INT5}$, $\overline{NMI}$, $\overline{ADTRG}$, $\overline{CTS0}$ to $\overline{CTS2}$, SCL, SDA, $CLK0$ to $CLK4$, $TA2_{OUT}$ to $TA4_{OUT}$, $RxD0$ to $RxD2$, $S_{IN3}$, $S_{IN4}$ | | | 0.2 | | 1.0 | V |
| $V_{T+}$-$V_{T-}$ | Hysteresis | $\overline{RESET}$ | | | 0.2 | | 2.2 | V |
| $I_{IH}$ | HIGH input current | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $X_{IN}$, $\overline{RESET}$, CNVss, M1, STARTB | | $V_I$=5V | | | 5.0 | µA |
| $I_{IL}$ | LOW input current | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $X_{IN}$, $\overline{RESET}$, CNVss, M1, STARTB | | $V_I$=0V | | | -5.0 | µA |
| $R_{PULLUP}$ | Pull-up resistance | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$ | | $V_I$=0V | 30 | 50 | 170 | kΩ |
| $R_{fXIN}$ | Feedback resistance | $X_{IN}$ | | | | 1.5 | | MΩ |
| $R_{fXCIN}$ | Feedback resistance | $X_{CIN}$ | | | | 15 | | MΩ |
| $V_{RAM}$ | RAM retention voltage | | | Stop mode | 2.0 | | | V |
| $V_{SYNCIN}$ | Sync voltage amplitude | | | | 0.3 | 0.6 | 1.2 | V |
| $V_{dat(text)}$ | Teletext data voltage amplitude | | | | 0.6 | 0.9 | 1.4 | V |
| $f_H$ | Horizontal synchronous signal frequency | | | | 14.6 | 15.625 | 17.0 | kHz |

Note 1: Referenced to $V_{CC}$=$V_{CC1}$=$V_{CC2}$=4.50 to 5.50 V, $V_{SS}$=0V at Topr = -20 to 70 °C, f(BCLK)=16 MHz unless otherwise specified.

**Table 16.8 Electrical Characteristics (2) (Note)**

$V_{CC1} = V_{CC2} = 3V$

| Symbol | Parameter | | | Measuring condition | Standard | | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | | | | Min. | Typ. | Max. | |
| $V_{OH}$ | HIGH output voltage | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$ | | $I_{OH}$ = -1 mA | $V_{CC2}$ - 0.5 | | $V_{CC}$ | V |
| | | $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_4$ | | $I_{OH}$ = -1 mA | $V_{CC1}$ - 0.5 | | $V_{CC}$ | V |
| $V_{OH}$ | HIGH output voltage | $X_{OUT}$ | HIGHPOWER | $I_{OH}$ = -0.1 mA | $V_{CC2}$ - 0.5 | | $V_{CC2}$ | V |
| | | | LOWPOWER | $I_{OH}$ = -50 µA | $V_{CC2}$ - 0.5 | | $V_{CC2}$ | |
| | HIGH output voltage | $X_{COUT}$ | HIGHPOWER | With no load applied | | 2.5 | | V |
| | | | LOWPOWER | With no load applied | | 1.6 | | |
| $V_{OL}$ | LOW output voltage | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$ | | $I_{OL}$ = 1 mA | | | 0.5 | V |
| $V_{OL}$ | LOW output voltage | $X_{OUT}$ | HIGHPOWER | $I_{OL}$ = 0.1 mA | | | 0.5 | V |
| | | | LOWPOWER | $I_{OL}$ = 50 µA | | | 0.5 | |
| | LOW output voltage | $X_{COUT}$ | HIGHPOWER | With no load applied | | 0 | | V |
| | | | LOWPOWER | With no load applied | | 0 | | |
| $V_{T+}$-$V_{T-}$ | Hysteresis | $TA0_{IN}$ to $TA4_{IN}$, $TB0_{IN}$ to $TB5_{IN}$, $\overline{INT0}$ to $\overline{INT5}$ $TA2_{OUT}$ to $TA4_{OUT}$ | | | 0.2 | | 0.8 | V |
| $V_{T+}$-$V_{T-}$ | Hysteresis | $\overline{RESET}$ | | | 0.2 | (0.7) | 1.8 | V |
| $I_{IH}$ | HIGH input voltage | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$ $X_{IN}$, $\overline{RESET}$, CNVss, M1, STARTB | | $V_I$ = 3 V | | | 4.0 | µA |
| $I_{IL}$ | HIGH input voltage | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$ $X_{IN}$, $\overline{RESET}$, CNVss, M1, STARTB | | $V_I$ = 0 V | | | -4.0 | µA |
| $R_{PULLUP}$ | Pull-up resistance | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$ | | $V_I$ = 0 V | 50 | 100 | 500 | kΩ |
| $R_{fXCIN}$ | Feedback resistance | $X_{IN}$ | | | | 3.0 | | MΩ |
| | Feedback resistance | $X_{CIN}$ | | | | 25 | | MΩ |

Note : Referenced to $V_{CC}$ = $V_{CC1}$ = $V_{CC2}$ = 3.0 V, $V_{SS}$ = 0 V at Topr = -20 to 70 °C, f ($X_{CIN}$) = 32kHz unless otherwise specified.
Use in single-chip mode and low power dissipation mode.

## Table 16.9 Electrical Characteristics (2) (Note 1)

| Symbol | Parameter | | Measuring condition | | Standard | | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | | | | Min. | Typ. | Max. | |
| $I_{CC}$ | Power supply current | In single-chip mode, the output pins are open and other pins are $V_{SS}$ | Mask ROM | f(BCLK)=16MHz, $V_{CC}$=5.0V | | 50 | 100 | mA |
| | | | Flash memory | f(BCLK)=16MHz, $V_{CC}$=5.0V | | 50 | 100 | mA |
| | | | Flash memory Program | f(BCLK)=16MHz, $V_{CC}$=5.0V | | 15 | | mA |
| | | | Flash memory Erase | f(BCLK)=16MHz, $V_{CC}$=5.0V | | 25 | | mA |
| | | | Mask ROM | f($X_{CIN}$)=32kHz, Low power dissipation mode, ROM(Note 3), (Note4) Vcc=5.0V | | 25 | | µA |
| | | | Flash memory | f(BCLK)=32kHz, Low power dissipation mode, RAM(Note 3), (Note4) Vcc=5.0V | | 25 | | µA |
| | | | | f(BCLK)=32kHz Low power dissipation mode, Flash memory(Note 3), (Note4) Vcc=5.0V | | 420 | | µA |
| | | | Mask ROM Flash memory | f(BCLK)=32kHz, Wait mode (Note 2), (Note4) Oscillation capacity High | | 7.5 | | µA |
| | | | | f(BCLK)=32kHz, Wait mode(Note 2), (Note4) Oscillation capacity Low Vcc=5.0V | | 5.0 | 10.0 | µA |
| | | | | f(BCLK)=32kHz, Wait mode (Note 2), (Note4) Oscillation capacity High Vcc=3.0V | | 6.0 | | µA |
| | | | | f(BCLK)=32kHz, Wait mode(Note 2), (Note4) Oscillation capacity Low Vcc=3.0V | | 2.0 | 8.0 | µA |
| | | | | Stop mode, (Note4) $T_{opr}$=25°C Vcc=5.0V | | 0.8 | 5.0 | µA |

Note 1: Referenced to $V_{CC1}$=$V_{CC2}$= 5V, $V_{SS}$=0V at Topr =25 °C, f(BCLK)=16MHz unless otherwise specified.
Note 2: With one timer operated using $f_{C32}$. (Slicer operation OFF)
Note 3: This indicates the memory in which the program to be executed exists.
Note 4: • All of $V_{DD2}$ is at the same potential level as $V_{CC2}$.
• Extension registers (addresses $00_{16}$ through $3F_{16}$) are set to the initial state.
• Inputs to the SYNCIN and CVIN pins are disabled.
• For current consumption reducing, set the level of $V_{SS}$ or $V_{CC}$ to the ports used in input mode.

## Table 16.10 Video signal input conditions (Note 1)

| Symbol | Parameter | Measuring condition | Standard | | | Unit |
|---|---|---|---|---|---|---|
| | | | Min | Typ. | Max. | |
| $V_{IN-cu}$ | Composite video signal input clamp voltage | Sync-chip voltage | | 1.0 | | V |

Note 1: Referenced to $V_{CC2}$ = 5.0 V at Topr = -20 to 70 °C unless otherwise specified.

Timing Requirements

**($V_{CC1}$ = $V_{CC2}$ = 5V, $V_{SS}$ = 0V, at Topr = – 20 to 70°C** unless otherwise specified)

**Table 16.11 External Clock Input ($X_{IN}$ input)**

$V_{CC1}$ = $V_{CC2}$ = 5V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_c$ | External clock input cycle time | 62.5 | | ns |
| $t_{w(H)}$ | External clock input HIGH pulse width | 30 | | ns |
| $t_{w(L)}$ | External clock input LOW pulse width | 30 | | ns |
| $t_r$ | External clock rise time | | 15 | ns |
| $t_f$ | External clock fall time | | 15 | ns |

**Table 16.12 Remote Control Pulse Input**

$V_{CC1}$ = $V_{CC2}$ = 5V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $T_{w(RMTH)}$ | $RMT_{IN}$ input HIGH pulse width | 61 | | µs |
| $T_{w(RMTL)}$ | $RMT_{IN}$ input LOW pulse width | 61 | | µs |

**Table 16.13 JUST CLOCK Input**

$V_{CC1}$ = $V_{CC2}$ = 5V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $T_{w(JSTH)}$ | $JST_{IN}$ input HIGH pulse width | 61 | | µs |
| $T_{w(JSTL)}$ | $JST_{IN}$ input LOW pulse width | 61 | | µs |

Timing Requirements

**($V_{CC1}$ = $V_{CC2}$ = 5V, $V_{SS}$ = 0V, at Topr = – 20 to 70°C unless otherwise specified)**

**Table 16.14 Timer A Input (Counter Input in Event Counter Mode)**

$V_{CC1}$ = $V_{CC2}$ = 5V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TA)}$ | TAiIN input cycle time | 100 | | ns |
| $t_{w(TAH)}$ | TAiIN input HIGH pulse width | 40 | | ns |
| $t_{w(TAL)}$ | TAiIN input LOW pulse width | 40 | | ns |

**Table 16.15 Timer A Input (Gating Input in Timer Mode)**

$V_{CC1}$ = $V_{CC2}$ = 5V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TA)}$ | TAiIN input cycle time | 400 | | ns |
| $t_{w(TAH)}$ | TAiIN input HIGH pulse width | 200 | | ns |
| $t_{w(TAL)}$ | TAiIN input LOW pulse width | 200 | | ns |

**Table 16.16 Timer A Input (External Trigger Input in One-shot Timer Mode)**

$V_{CC1}$ = $V_{CC2}$ = 5V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TA)}$ | TAiIN input cycle time | 200 | | ns |
| $t_{w(TAH)}$ | TAiIN input HIGH pulse width | 100 | | ns |
| $t_{w(TAL)}$ | TAiIN input LOW pulse width | 100 | | ns |

**Table 16.17 Timer A Input (External Trigger Input in Pulse Width Modulation Mode)**

$V_{CC1}$ = $V_{CC2}$ = 5V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{w(TAH)}$ | TAiIN input HIGH pulse width | 100 | | ns |
| $t_{w(TAL)}$ | TAiIN input LOW pulse width | 100 | | ns |

**Table 16.18 Timer A Input (Counter Increment/decrement Input in Event Counter Mode)**

$V_{CC1}$ = $V_{CC2}$ = 5V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(UP)}$ | TAiOUT input cycle time | 2000 | | ns |
| $t_{w(UPH)}$ | TAiOUT input HIGH pulse width | 1000 | | ns |
| $t_{w(UPL)}$ | TAiOUT input LOW pulse width | 1000 | | ns |
| $t_{su(UP-TIN)}$ | TAiOUT input setup time | 400 | | ns |
| $t_{h(TIN-UP)}$ | TAiOUT input hold time | 400 | | ns |

Timing Requirements

**($V_{CC1}$ = $V_{CC2}$ = 5V, $V_{SS}$ = 0V, at Topr = – 20 to 70°C unless otherwise specified)**

### Table 16.19 Timer B Input (Counter Input in Event Counter Mode)

$V_{CC1}$ = $V_{CC2}$ = 5V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TB)}$ | TBiIN input cycle time (counted on one edge) | 100 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width (counted on one edge) | 40 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width (counted on one edge) | 40 | | ns |
| $t_{c(TB)}$ | TBiIN input HIGH pulse width (counted on both edges) | 200 | | ns |
| $t_{w(TBH)}$ | TBiIN input LOW pulse width (counted on both edges) | 80 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width (counted on both edges) | 80 | | ns |

### Table 16.20 Timer B Input (Pulse Period Measurement Mode))

$V_{CC1}$ = $V_{CC2}$ = 5V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TB)}$ | TBiIN input cycle time | 400 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width | 200 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width | 200 | | ns |

### Table 16.21 Timer B Input (Pulse Width Measurement Mode))

$V_{CC1}$ = $V_{CC2}$ = 5V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TB)}$ | TBiIN input cycle time | 400 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width | 200 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width | 200 | | ns |

### Table 16.22 A/D Trigger Input

$V_{CC1}$ = $V_{CC2}$ = 5V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(AD)}$ | $\overline{ADTRG}$ input cycle time (trigger able minimum) | 1000 | | ns |
| $t_{w(ADL)}$ | $\overline{ADTRG}$ input LOW pulse width | 125 | | ns |

### Table 16.23 Serial I/O

$V_{CC1}$ = $V_{CC2}$ = 5V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(CK)}$ | CLKi input cycle time | 200 | | ns |
| $t_{w(CKH)}$ | CLKi input HIGH pulse width | 100 | | ns |
| $t_{w(CKL)}$ | CLKi input LOW pulse width | 100 | | ns |
| $t_{d(C-Q)}$ | TxDi output delay time | | 80 | ns |
| $t_{h(C-Q)}$ | TxDi hold time | 0 | | ns |
| $t_{su(D-C)}$ | RxDi input setup time | 30 | | ns |
| $t_{h(C-D)}$ | RxDi input hold time | 90 | | ns |

### Table 16.24 External Interrupt INTi Input

$V_{CC1}$ = $V_{CC2}$ = 5V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{w(INH)}$ | $\overline{INTi}$ input HIGH pulse width | 250 | | ns |
| $t_{w(INL)}$ | $\overline{INTi}$ input LOW pulse width | 250 | | ns |

Timing Requirements

**($V_{CC1}$ = $V_{CC2}$ = 3V, $V_{SS}$ = 0V, at Topr = – 20 to 70°C unless otherwise specified)**

**Table 16.25 External Clock Input ($X_{IN}$ Input)**

$V_{CC1}$ = $V_{CC2}$ = 3V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_c$ | External clock input cycle time | 100 | | ns |
| $t_{w(H)}$ | External clock input HIGH pulse width | 40 | | ns |
| $t_{w(L)}$ | External clock input LOW pulse width | 40 | | ns |
| $t_r$ | External clock rise time | | 18 | ns |
| $t_f$ | External clock fall time | | 18 | ns |

Timing Requirements

**($V_{CC1}$ = $V_{CC2}$ = 3V, $V_{SS}$ = 0V, at Topr = – 20 to 70°C unless otherwise specified)**

**Table 16.26 Timer A Input (Counter Input in Event Counter Mode)**

$V_{CC1}$ = $V_{CC2}$ = 3V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TA)}$ | $TAi_{IN}$ input cycle time | 150 | | ns |
| $t_{w(TAH)}$ | $TAi_{IN}$ input HIGH pulse width | 60 | | ns |
| $t_{w(TAL)}$ | $TAi_{IN}$ input LOW pulse width | 60 | | ns |

**Table 16.27 Timer A Input (Gating Input in Timer Mode)**

$V_{CC1}$ = $V_{CC2}$ = 3V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TA)}$ | $TAi_{IN}$ input cycle time | 600 | | ns |
| $t_{w(TAH)}$ | $TAi_{IN}$ input HIGH pulse width | 300 | | ns |
| $t_{w(TAL)}$ | $TAi_{IN}$ input LOW pulse width | 300 | | ns |

**Table 16.28 Timer A Input (External Trigger Input in One-shot Timer Mode)**

$V_{CC1}$ = $V_{CC2}$ = 3V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TA)}$ | $TAi_{IN}$ input cycle time | 300 | | ns |
| $t_{w(TAH)}$ | $TAi_{IN}$ input HIGH pulse width | 150 | | ns |
| $t_{w(TAL)}$ | $TAi_{IN}$ input LOW pulse width | 150 | | ns |

**Table 16.29 Timer A Input (External Trigger Input in Pulse Width Modulation Mode)**

$V_{CC1}$ = $V_{CC2}$ = 3V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{w(TAH)}$ | $TAi_{IN}$ input HIGH pulse width | 150 | | ns |
| $t_{w(TAL)}$ | $TAi_{IN}$ input LOW pulse width | 150 | | ns |

**Table 16.30 Timer A Input (Counter Increment/decrement Input in Event Counter Mode)**

$V_{CC1}$ = $V_{CC2}$ = 3V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(UP)}$ | $TAi_{OUT}$ input cycle time | 3000 | | ns |
| $t_{w(UPH)}$ | $TAi_{OUT}$ input HIGH pulse width | 1500 | | ns |
| $t_{w(UPL)}$ | $TAi_{OUT}$ input LOW pulse width | 1500 | | ns |
| $t_{su(UP-TIN)}$ | $TAi_{OUT}$ input setup time | 600 | | ns |
| $t_{h(TIN-UP)}$ | $TAi_{OUT}$ input hold time | 600 | | ns |

Timing Requirements

**($V_{CC1}$ = $V_{CC2}$ = 3V, $V_{SS}$ = 0V, at Topr = – 20 to 70°C unless otherwise specified)**

**Table 16.31 Timer B Input (Counter Input in Event Counter Mode)**

$V_{CC1}$ = $V_{CC2}$ = 3V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TB)}$ | TBiIN input cycle time (counted on one edge) | 150 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width (counted on one edge) | 60 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width (counted on one edge) | 60 | | ns |
| $t_{c(TB)}$ | TBiIN input cycle time (counted on both edges) | 300 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width (counted on both edges) | 120 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width (counted on both edges) | 120 | | ns |

**Table 16.32 Timer B Input (Pulse Period Measurement Mode)**

$V_{CC1}$ = $V_{CC2}$ = 3V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TB)}$ | TBiIN input cycle time | 600 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width | 300 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width | 300 | | ns |

**Table 16.33 Timer B Input (Pulse Period Measurement Mode)**

$V_{CC1}$ = $V_{CC2}$ = 3V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TB)}$ | TBiIN input cycle time | 600 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width | 300 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width | 300 | | ns |

**Table 16.34 Serial I/O (Pulse Period Measurement Mode)**

$V_{CC1}$ = $V_{CC2}$ = 3V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(CK)}$ | CLKi input cycle time | 300 | | ns |
| $t_{w(CKH)}$ | CLKi input HIGH pulse width | 150 | | ns |
| $t_{w(CKL)}$ | CLKi input LOW pulse width | 150 | | ns |
| $t_{d(C-Q)}$ | TxDi output delay time | | 160 | ns |
| $t_{h(C-Q)}$ | TxDi hold time | 0 | | ns |
| $t_{su(D-C)}$ | RxDi input setup time | 70 | | ns |
| $t_{h(C-D)}$ | RxDi input hold time | 90 | | ns |

**Table 16.35 External Interrupt INTi Input (Pulse Period Measurement Mode)**

$V_{CC1}$ = $V_{CC2}$ = 3V

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{w(INH)}$ | $\overline{INTi}$ input HIGH pulse width | 380 | | ns |
| $t_{w(INL)}$ | $\overline{INTi}$ input LOW pulse width | 380 | | ns |

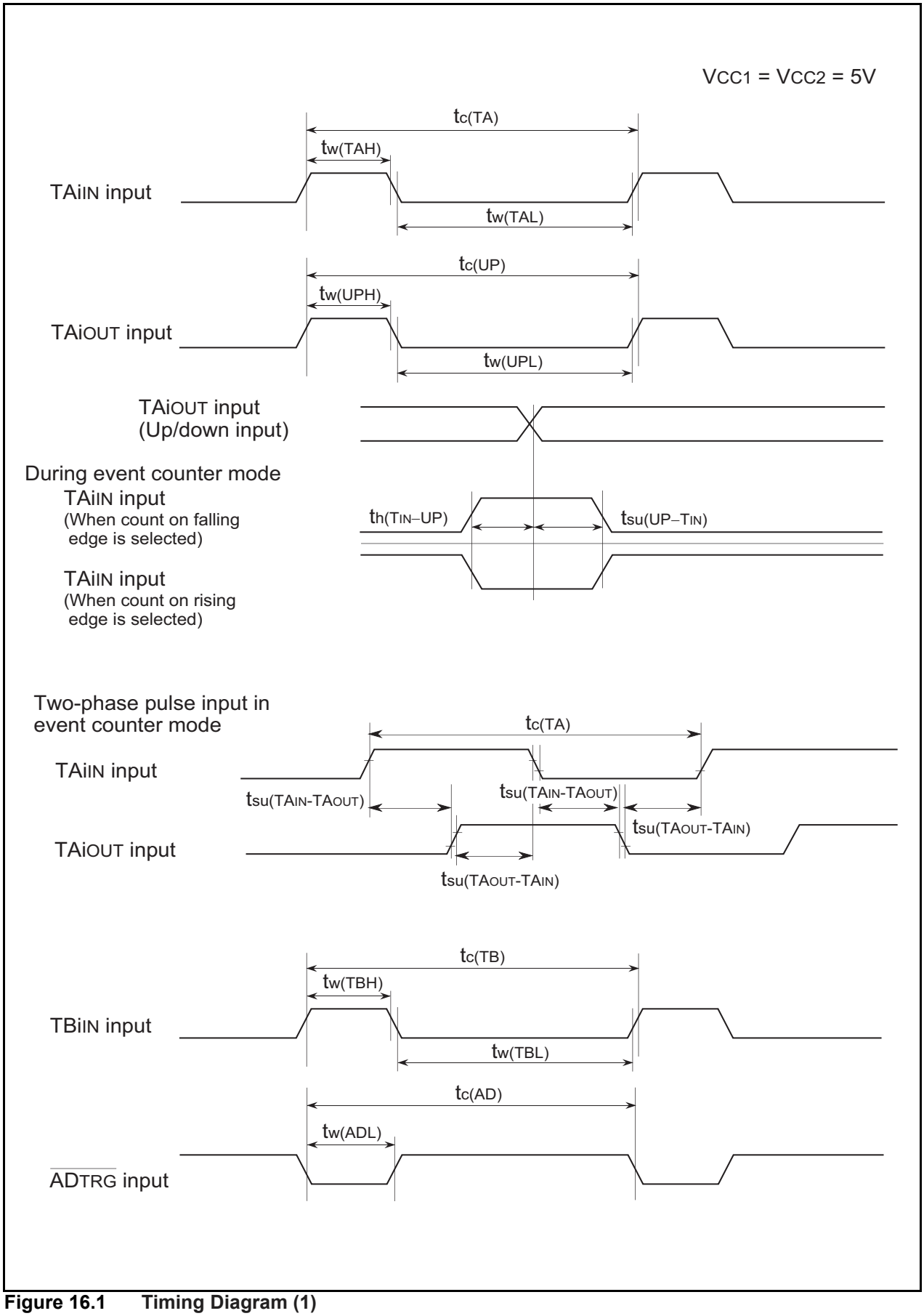



**Figure 16.1 Timing Diagram (1)**

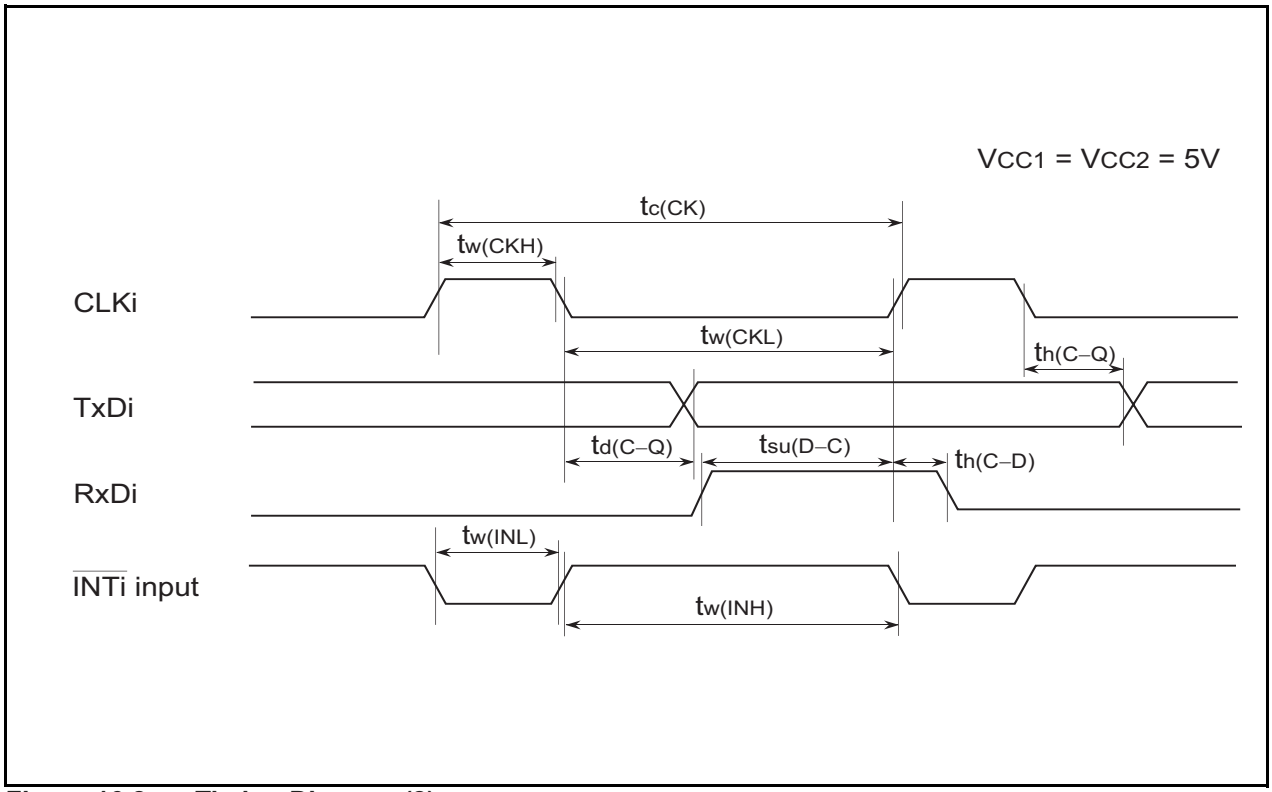



**Figure 16.2 Timing Diagram (2)**

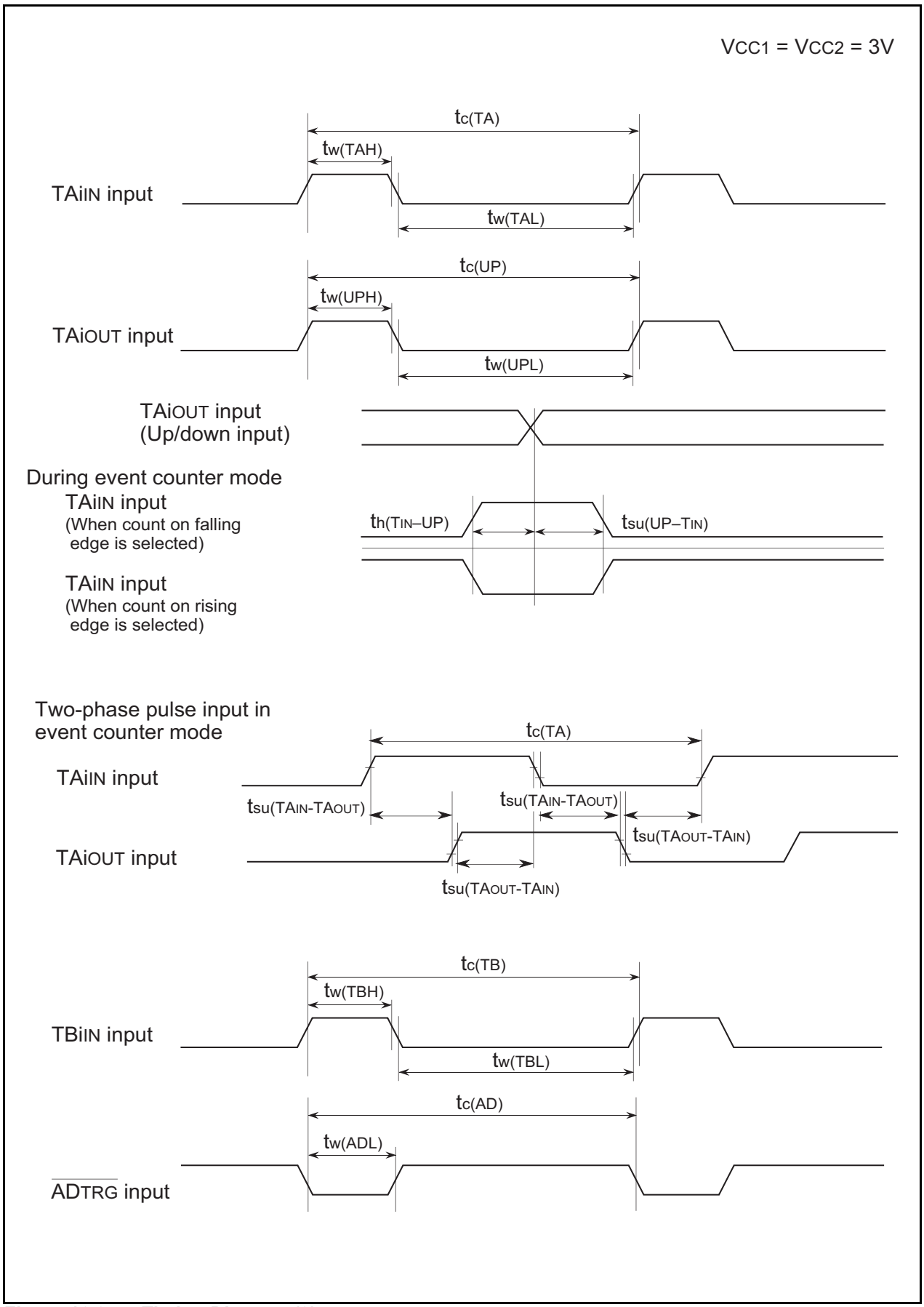



**Figure 16.3 Timing Diagram (3)**

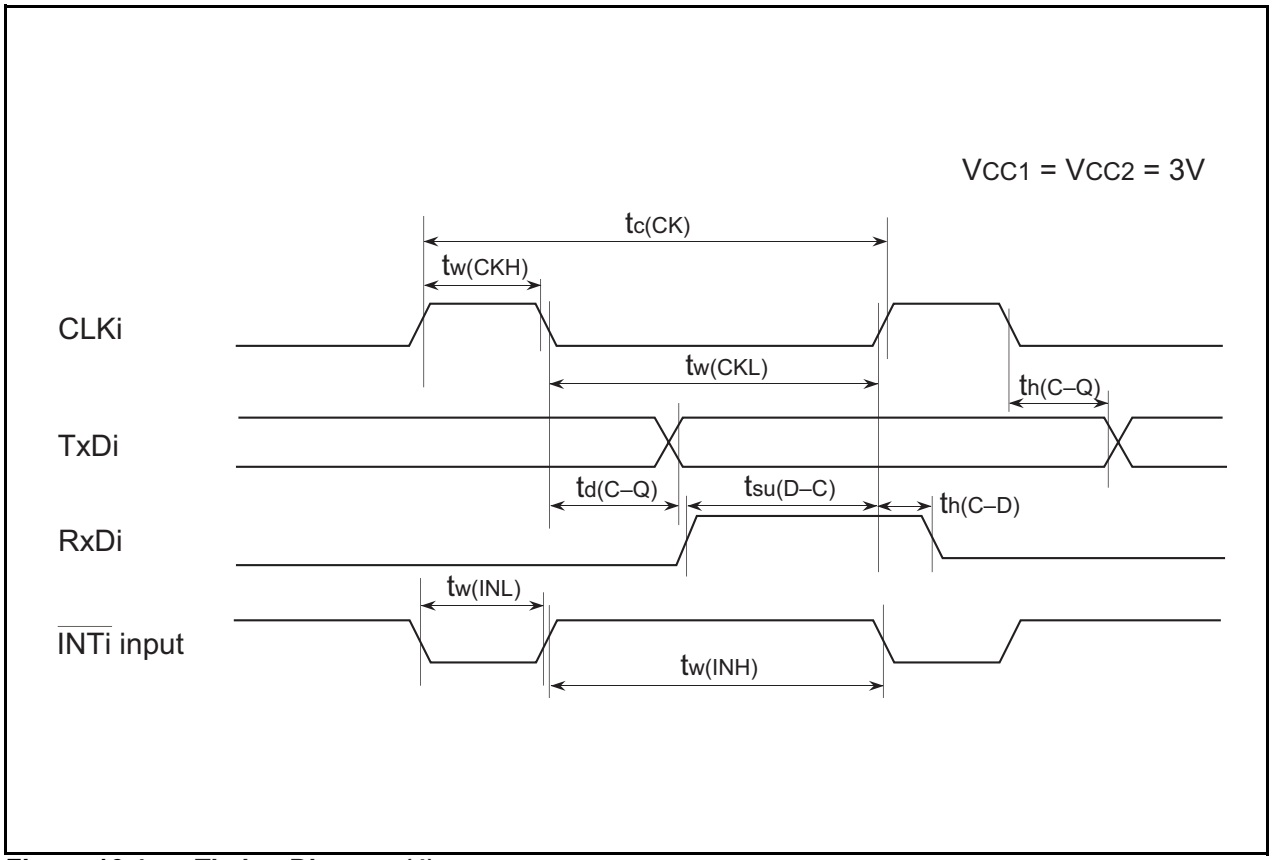



**Figure 16.4 Timing Diagram (4)**

# 17. Flash Memory Version

## 17.1 Flash Memory Performance

The flash memory version is functionally the same as the mask ROM version except that it internally contains flash memory.
The flash memory version has three modes−CPU rewrite, standard serial input/output, and parallel input/output modes−in which its internal flash memory can be operated on.
Table 17.1 shows the outline performance of flash memory version (see Table 1.1 for the items not listed in Table 17.1.).

**Table 17.1 Flash Memory Version Specifications**

| Item | | Specification |
|---|---|---|
| Flash memory operating mode | | 3 modes (CPU rewrite, standard serial I/O, parallel I/O) |
| Erase block | User ROM area | See Figure 17.1 |
| | Boot ROM area | 1 block (4 Kbytes) (Note 1) |
| Method for program | | In units of word |
| Method for erasure | | Block erase |
| Program, erase control method | | Program and erase controlled by software command |
| Protect method | | Protected for each block by lock bit |
| Number of commands | | 7 commands |
| Number of program and erasure | | 100 times |
| Data Retention | | 10 years |
| ROM code protection | | Parallel I/O and standard serial I/O modes are supported. |

Note 1: The boot ROM area contains a standard serial I/O mode rewrite control program which is stored in it when shipped from the factory. This area can only be rewritten in parallel input/output mode.

**Table 17.2 Flash Memory Rewrite Modes Overview**

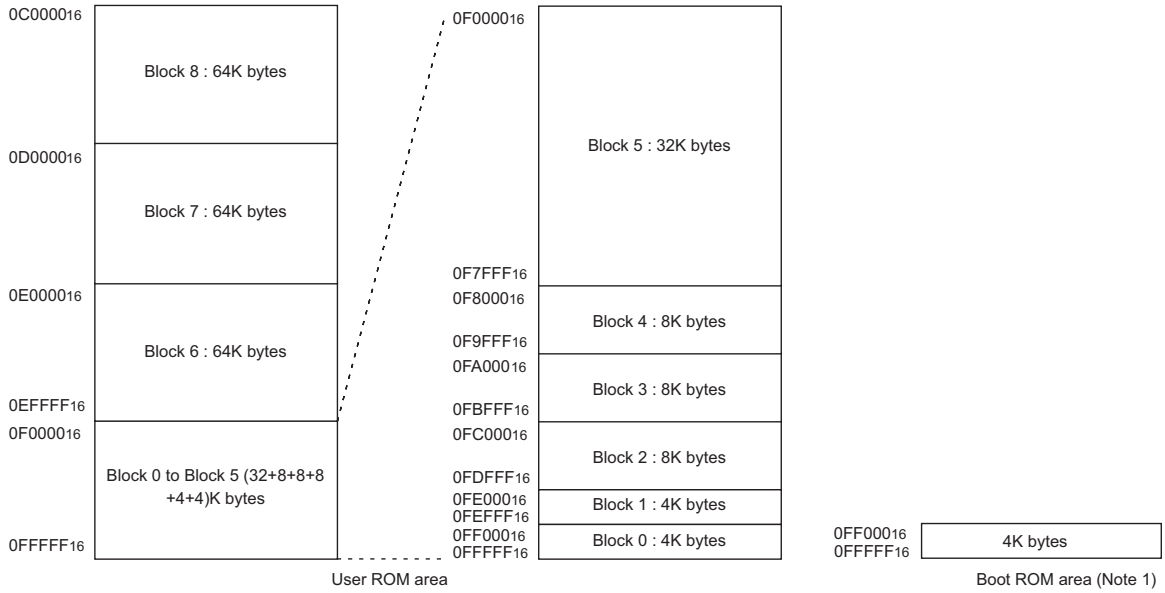
| Flash memory rewrite mode | CPU rewrite mode (Note 1) | Standard serial I/O mode | Parallel I/O mode |
|---|---|---|---|
| Function | The user ROM area is rewritten by executing software commands from the CPU.<br>EW0 mode:<br>Can be rewritten in any area other than the flash memory (Note 2)<br>EW1 mode:<br>Can be rewritten in the flash memory | The user ROM area is rewritten by using a dedicated serial programmer.<br>Standard serial I/O mode 1:<br>Clock sync serial I/O<br>Standard serial I/O mode 2:<br>UART | The boot ROM and user ROM areas are rewritten by using a dedicated parallel programmer. |
| Areas which can be rewritten | User ROM area | User ROM area | User ROM area<br>Boot ROM area |
| Operation mode | Single chip mode<br>Boot mode (EW0 mode) | Boot mode | Parallel I/O mode |
| ROM programmer | None | Serial programmer | Parallel programmer |

Note 1: Bit 3 of processor mode register 1 remains set to "1" while the FMR0 register FMR01 bit = 1 (CPU rewrite mode enabled).
Bit 3 of processor mode register 1 is reverted to its original value by clearing the FMR01 bit to "0" (CPU rewrite mode disabled). However, if bit 3 of processor mode register 1 is changed during CPU rewrite mode, its changed value is not reflected until after the FMR01 bit is cleared to "0".

Note 2: When in CPU rewrite mode, bit 0 and bit 3 in the PM1 register are set to "1". The rewrite control program can only be executed in the internal RAM.

## 17.2 Memory Map

The ROM in the flash memory version is separated between a user ROM area and a boot ROM area.
Figure 17.1 shows the block diagram of flash momoery.
The user ROM area is divided into several blocks, each of which can individually be protected (locked) against programming or erasure. The user ROM area can be rewritten in all of CPU rewrite, standard serial input/output, and parallel input/output modes.
The boot ROM area is located at addresses that overlap the user ROM area, and can only be rewritten in parallel input/output mode. After a hardware reset that is performed by applying a high-level signal to the CNVSS and P50 pins and a low-level signal to the M1 pin, the program in the boot ROM area is executed.
After a hardware reset that is performed by applying a low-level signal to the CNVSS pin, the program in the user ROM area is executed (but the boot ROM area cannot be read).




Note 1: The boot ROM area can only be rewritten in parallel input/output mode.
Note 2: To specify a block, use an even address in that block.
Note 3: Shown here is a block diagram during single-chip mode.

**Figure 17.1 Flash Memory Block Diagram**

## 17.3 Boot Mode

After a hardware reset which is performed by applying a low-level signal to the M1 pin and a high-level signal to the CNVSS and P50 pins, the microcomputer is placed in boot mode, thereby executing the program in the boot ROM area.
During boot mode, the boot ROM and user ROM areas are switched over by the FMR05 bit in the FMR0 register.
The boot ROM area contains a standard serial input/output mode based rewrite control program which was stored in it when shipped from the factory.
The boot ROM area can be rewritten in parallel input/output mode. Prepare an EW0 mode based rewrite control program and write it in the boot ROM area, and the flash memory can be rewritten as suitable for the system.

## 17.4 Functions To Prevent Flash Memory from Rewriting

To prevent the flash memory from being read or rewritten easily, parallel input/output mode has a ROM code protect and standard serial input/output mode has an ID code check function.

### 17.4.1 ROM Code Protect Function

The ROM code protect function inhibits the flash memory from being read or rewritten during parallel input/output mode. Figure 17.2 shows the ROMCP register.
The ROMCP register is located in the user ROM area. The ROMCP1 bit consists of two bits. The ROM code protect function is enabled by clearing one or both of two ROMCP1 bits to “0” when the ROMCR bits are not ‘$00_2$,’ with the flash memory thereby protected against reading or rewriting. Conversely, when the ROMCR bits are ‘$00_2$’ (ROM code protect removed), the flash memory can be read or rewritten. Once the ROM code protect function is enabled, the ROMCR bits cannot be changed during parallel input/output mode. Therefore, use standard serial input/output or other modes to rewrite the flash memory.

### 17.4.2 ID Code Check Function

Use this function in standard serial input/output mode. Unless the flash memory is blank, the ID codes sent from the programmer and the ID codes written in the flash memory are compared to see if they match. If the ID codes do not match, the commands sent from the programmer are not accepted. The ID code consists of 8-bit data, the areas of which, beginning with the first byte, are $0FFFDF_{16}$, $0FFFE3_{16}$, $0FFFEB_{16}$, $0FFFEF_{16}$, $0FFFF3_{16}$, $0FFFF7_{16}$, and $0FFFFB_{16}$. Prepare a program in which the ID codes are preset at these addresses and write it in the flash memory.

ROM code protect control address

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 | Symbol | Address | Value when shipped |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 1 | 1 | 1 | 1 | ROMCP | 0FFFFF16 | FF16 (Note 4) |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| ——— | Reserved bit | Set this bit to "1" | RW |
| ——— | Reserved bit | Set this bit to "1" | RW |
| ——— | Reserved bit | Set this bit to "1" | RW |
| ——— | Reserved bit | Set this bit to "1" | RW |
| ROMCR | ROM code protect reset bit (Note 2, Note 4) | b5 b4<br>00: Removes protect<br>01: } Enables ROMCP1 bit<br>10: } Enables ROMCP1 bit<br>11: } Enables ROMCP1 bit | RW<br>RW |
| ROMCP1 | ROM code protect level 1 set bit (Note 1, Note 3, Note 4) | b7 b6<br>00: } Protect enabled<br>01: } Protect enabled<br>10: } Protect enabled<br>11: Protect disabled | RW<br>RW |

Note 1: If the ROMCR bits are set to other than "002" and the ROMCP1 bits are set to other than "112" (ROM code protect enabled), the flash memory is disabled against reading and rewriting in parallel input/output mode.
Note 2: If the ROMCR bits are set to "002," ROM code protect level 1 is removed. However, because the ROMCR bits cannot be modified during parallel input/output mode, they need to be modified in standard serial input/output or other modes.
Note 3: The ROMCP1 bits are effective when the ROMCR bits are "012," "102," or "112."
Note 4: Once any of these bits is cleared to "0", it cannot be set back to "1". If a memory block that contains the ROMCP register is erased, the ROMCP register is set to "FF16."

**Figure 17.2 ROMCP Register**

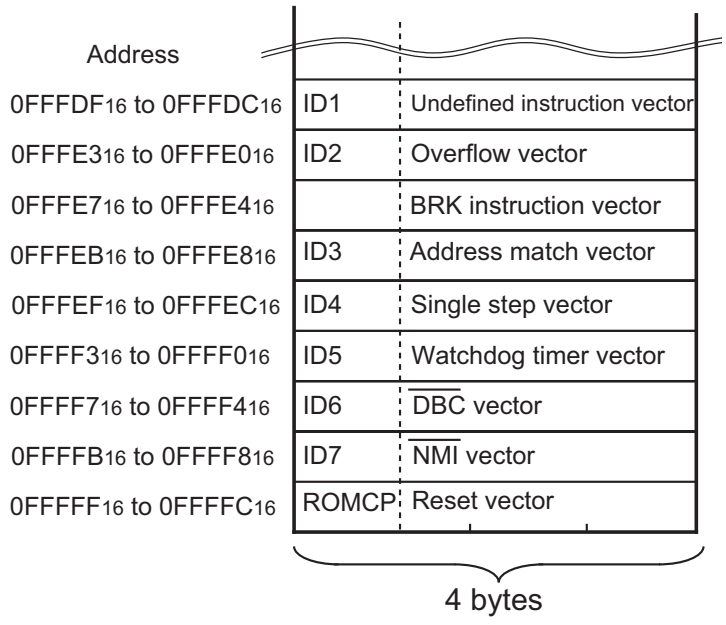



**Figure 17.3 Address for ID Code Stored**

## 17.5 CPU Rewrite Mode

In CPU rewrite mode, the user ROM area can be rewritten by executing software commands from the CPU. Therefore, the user ROM area can be rewritten directly while the microcomputer is mounted on-board without having to use a ROM programmer, etc.

In CPU rewrite mode, only the user ROM area shown in Figure 17.1 can be rewritten and the boot ROM area cannot be rewritten. Make sure the Program and the Block Erase commands are executed only on each block in the user ROM area.

During CPU rewrite mode, the user ROM area be operated on in either Erase Write 0 (EW0) mode or Erase Write 1 (EW1) mode. Table 17.3 lists the differences between Erase Write 0 (EW0) and Erase Write 1 (EW1) modes.

**Table 17.3 EW0 Mode and EW1 Mode**

| Item | EW0 mode | EW1 mode |
|---|---|---|
| Operation mode | • Single chip mode<br>• Boot mode | Single chip mode |
| Areas in which a rewrite control program can be located | • User ROM area<br>• Boot ROM area | User ROM area |
| Areas in which a rewrite control program can be executed | Must be transferred to any area other than the flash memory (RAM) before being executed (Note 2) | Can be executed directly in the user ROM area |
| Areas which can be rewritten | User ROM area | User ROM area<br>However, this does not include the area in which a rewrite control program exists |
| Software command limitations | None | • Program, Block Erase command<br>Cannot be executed on any block in which a rewrite control program exists<br>• Read Status Register command<br>Cannot be executed |
| Modes after Program or Erase | Read Status Register mode | Read Array mode |
| CPU status during Auto Write and Auto Erase | Operating | Hold state (I/O ports retain the state in which they were before the command was executed)(Note 1) |
| Flash memory status detection | • Read the FMR0 register's FMR00, FMR06, and FMR07 bits in a program<br>• Execute the Read Status Register command to read the status register's SR7, SR5, and SR4 flags. | Read the FMR0 register's FMR00, FMR06, and FMR07 bits in a program |

Note 1: Make sure no interrupts (except $\overline{\text{NMI}}$ and watchdog timer interrupts) and DMA transfers will occur.
Note 2: When in CPU rewrite mode, bit 0 and bit 3 in the PM1 register are set to "1". The rewrite control program can only be executed in the internal RAM.

### 17.5.1 EW0 Mode

The microcomputer is placed in CPU rewrite mode by setting the FMR0 register's FMR01 bit to "1" (CPU rewrite mode enabled), ready to accept commands. In this case, because the FMR1 register's FMR11 bit = 0, EW0 mode is selected. The FMR01 bit can be set to "1" by writing "0" and then "1" in succession.
Use software commands to control program and erase operations. Read the FMR0 register or status register to check the status of program or erase operation at completion.

### 17.5.2 EW1 Mode

EW1 mode is selected by setting FMR11 bit to "1" (by writing "0" and then "1" in succession) after setting the FMR01 bit to "1" (by writing "0" and then "1" in succession).
Read the FMR0 register to check the status of program or erase operation at completion. The status register cannot be read during EW1 mode.
Figure 17.4 shows the FMR0 and FMR1 registers.
Registers FMR0 and FMR1 are shown in Figure 17.4.

### FMR00 Bit

This bit indicates the operating status of the flash memory. The bit is "0" when the Program, Erase, or Lock Bit program is running; otherwise, the bit is "1".

### FMR01 Bit

The microcomputer is made ready to accept commands by setting the FMR01 bit to "1" (CPU rewrite mode). During boot mode, make sure the FMR05 bit also is "1" (user ROM area access).

### FMR02 Bit

The lock bit set for each block can be disabled by setting the FMR02 bit to "1" (lock bit disabled). (Refer to the description of the data protect function.) The lock bits set are enabled by setting the FMR02 bit to "0".
The FMR02 bit only disables the lock bit function and does not modify the lock bit data (lock bit status flag). However, if the Erase command is executed while the FMR02 bit is set to "1", the lock bit data changes state from "0" (locked) to "1" (unlocked) after Erase is completed.

### FMSTP Bit

This bit is provided for initializing the flash memory control circuits, as well as for reducing the amount of current consumed in the flash memory. The internal flash memory is disabled against access by setting the FMSTP bit to "1". Therefore, make sure the FMSTP bit is modified in other than the flash memory.
In the following cases, set the FMSTP bit to "1":

- . When flash memory access resulted in an error while erasing or programming in EW0 mode (FMR00 bit not reset to "1" (ready))
- . When entering low power mode

Figure 17.7 shows a flow chart to be followed before and after entering low power mode.
Note that when going to stop or wait mode, the FMR0 register does not need to be set because the power for the internal flash memory is automatically turned off and is turned back on again after returning from stop or wait mode.

### FMR05 Bit

This bit switches between the boot ROM and user ROM areas during boot mode. Set this bit to "0" when accessing the boot ROM area (for read) or "1" (user ROM access) when accessing the user ROM area (for read, write, or erase).

### FMR06 Bit

This is a read-only bit indicating the status of auto program operation. The bit is set to "1" when a program error occurs; otherwise, it is cleared to "0". For details, refer to the description of the full status check.

### FMR07 Bit

This is a read-only bit indicating the status of auto erase operation. The bit is set to "1" when an erase error occurs; otherwise, it is cleared to "0". For details, refer to the description of the full status check.

Figure 17.5 and 17.6 show the setting and resetting of EW0 mode and EW1 mode, respectively.

### FMR11 Bit

Setting this bit to "1" places the microcomputer in EW1 mode.

### FMR16 Bit

This is a read-only bit indicating the execution result of the Read Lock Bit Status command.

Flash memory control register 0

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | 0 | | | | |

| Symbol | Address | After reset |
|---|---|---|
| FMR0 | $01B7_{16}$ | $XX000001_2$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| FMR00 | RY/$\overline{BY}$ status flag | 0: Busy (being written or erased)<br>1: Ready | RO |
| FMR01 | CPU rewrite mode select bit (Note 1) | 0: Disables CPU rewrite mode<br>1: Enables CPU rewrite mode | RW |
| FMR02 | Lock bit disable select bit (Note 2) | 0: Enables lock bit<br>1: Disables lock bit | RW |
| FMSTP | Flash memory stop bit (Note 3, Note 5)) | 0: Enables flash memory operation<br>1: Stops flash memory operation (placed in low power mode, flash memory initialized) | RW |
| $\overline{(b4)}$ | Reserved bit | Must always be set to “0” | RW |
| FMR05 | User ROM area select bit (Note 3) (Effective in only boot mode) | 0: Boot ROM area is accessed<br>1: User ROM area is accessed | RW |
| FMR06 | Program status flag (Note 4) | 0: Terminated normally<br>1: Terminated in error | RO |
| FMR07 | Erase status flag (Note 4) | 0: Terminated normally<br>1: Terminated in error | RO |

Note 1: To set this bit to “1”, write “0” and then “1” in succession. Make sure no interrupts or DMA transfers will occur before writing “1” after writing “0”.
Write to this bit when the $\overline{NMI}$ pin is in the high state. Also, while in EW0 mode, modify this bit in other than the flash memory.
Note 2: To set this bit to “1”, write “0” and then “1” in succession when the FMR01 bit = 1. Make sure no interrupts or no DMA transfers will occur before writing “1” after writing “0”.
Note 3: modify this bit in other than the flash memory.
Note 4: This flag is cleared to “0” by executing the Clear Status command.
Note 5: Effective when the FMR01 bit = 1 (CPU rewrite mode). If the FMR01 bit = 0, although the FMSTP bit can be set to “1” by writing “1” in a program, the flash memory is neither placed in low power mode nor initialized.
Note 6: This status includes writing or reading with the Lock Bit Program or Read Lock Bit Status command.

Flash memory control register 1

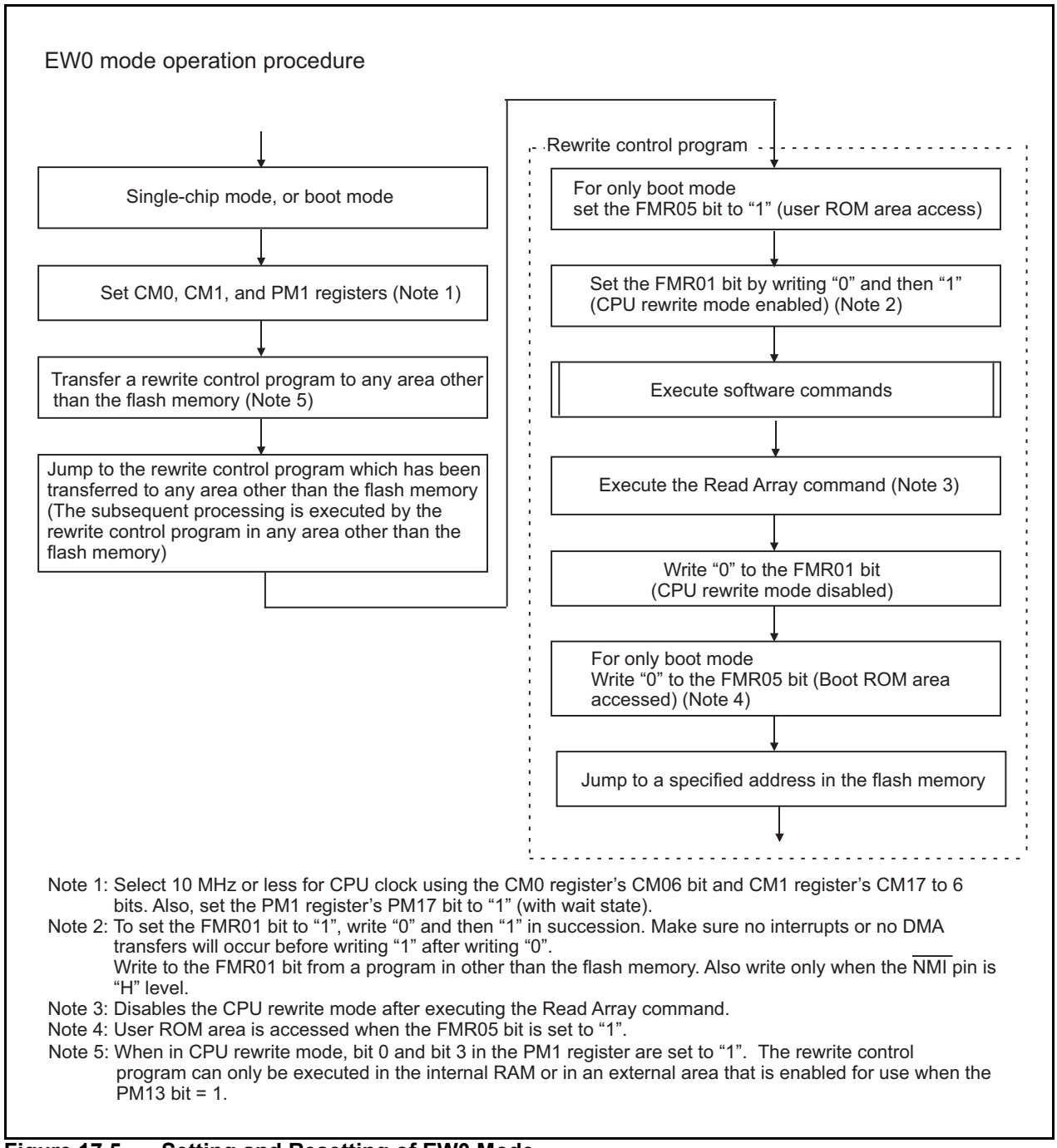
| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| 0 | | 0 | 0 | | | | 0 |

| Symbol | Address | After reset |
|---|---|---|
| FMR1 | $01B5_{16}$ | $0X00XX0X_2$ |

| Bit symbol | Bit name | Function | RW |
|---|---|---|---|
| $\overline{(b0)}$ | Reserved bit | The value in this bit when read is indeterminate. | RO |
| FMR11 | EW1 mode select bit (Note) | 0: EW0 mode<br>1: EW1 mode | RW |
| $\overline{(b3\text{-}b2)}$ | Reserved bit | The value in this bit when read is indeterminate. | RO |
| $\overline{(b5\text{-}b4)}$ | Reserved bit | Must always be set to “0” | RW |
| FMR16 | Lock bit status flag | 0: Lock<br>1: Unlock | RO |
| $\overline{(b7)}$ | Reserved bit | Must always be set to “0” | RW |

Note : To set this bit to “1”, write “0” and then “1” in succession when the FMR01 bit = 1. Make sure no interrupts or no DMA transfers will occur before writing “1” after writing “0”. Write this bit in the state the $\overline{NMI}$ pin = “H”. The FMR01 and FMR11 bits both are cleared to “0” by setting the FMR01 bit to “0”.

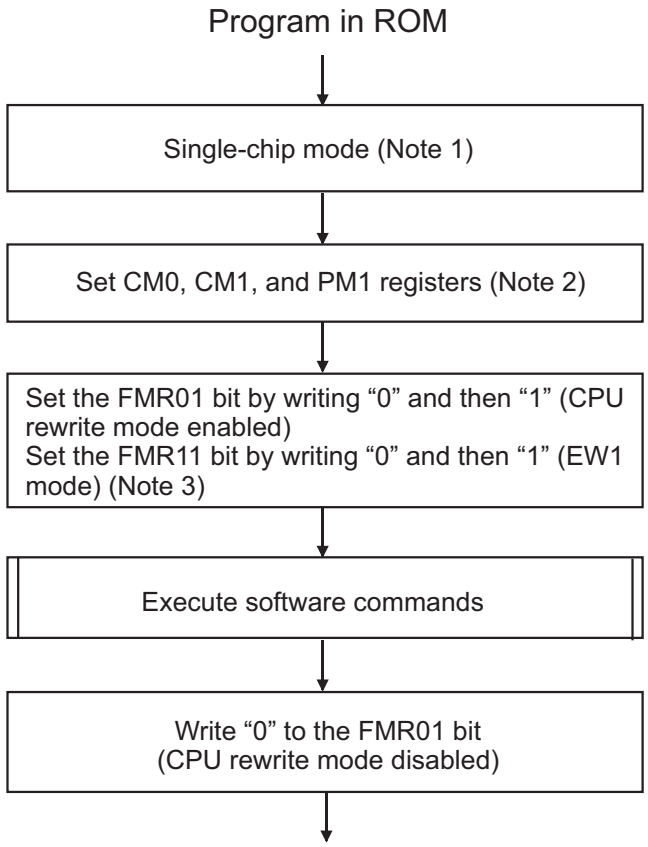
**Figure 17.4 FMR0 and FMR1 Registers**




Note 1: Select 10 MHz or less for CPU clock using the CM0 register's CM06 bit and CM1 register's CM17 to 6 bits. Also, set the PM1 register's PM17 bit to "1" (with wait state).
Note 2: To set the FMR01 bit to "1", write "0" and then "1" in succession. Make sure no interrupts or no DMA transfers will occur before writing "1" after writing "0".
Write to the FMR01 bit from a program in other than the flash memory. Also write only when the $\overline{\text{NMI}}$ pin is "H" level.
Note 3: Disables the CPU rewrite mode after executing the Read Array command.
Note 4: User ROM area is accessed when the FMR05 bit is set to "1".
Note 5: When in CPU rewrite mode, bit 0 and bit 3 in the PM1 register are set to "1". The rewrite control program can only be executed in the internal RAM or in an external area that is enabled for use when the PM13 bit = 1.

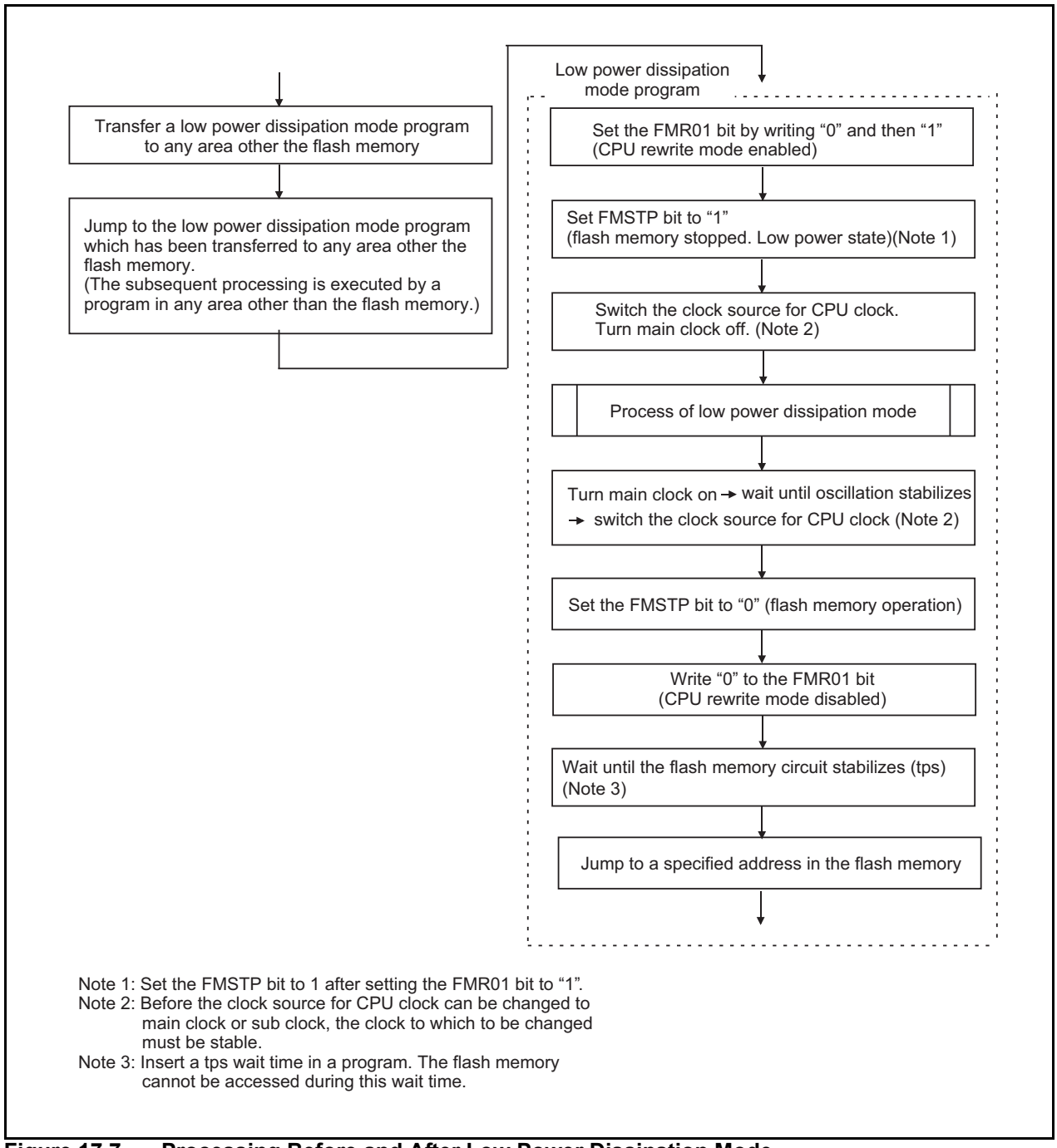
**Figure 17.5 Setting and Resetting of EW0 Mode**

EW1 mode operation procedure




Note 1: In EW1 mode, do not set the microcomputer in boot mode.
Note 2: Select 10 MHz or less for CPU clock using the CM0 register’s CM06 bit and CM1 register’s CM17 to 6 bits. Also, set the PM1 register’s PM17 bit to “1” (with wait state).
Note 3: To set the FMR01 bit to “1”, write “0” and then “1” in succession. Make sure no interrupts or no DMA transfers will occur before writing “1” after writing “0”. Write to the FMR01 bit from a program in other than the flash memory. Also write only when the $\overline{\text{NMI}}$ pin is “H” level.

**Figure 17.6 Setting and Resetting of EW1 Mode**




Note 1: Set the FMSTP bit to 1 after setting the FMR01 bit to "1".
Note 2: Before the clock source for CPU clock can be changed to main clock or sub clock, the clock to which to be changed must be stable.
Note 3: Insert a tps wait time in a program. The flash memory cannot be accessed during this wait time.

**Figure 17.7 Processing Before and After Low Power Dissipation Mode**

### 17.5.3 Precautions on CPU Rewrite Mode

Described below are the precautions to be observed when rewriting the flash memory in CPU rewrite mode.

(1) Operation Speed

Before entering CPU rewrite mode (EW0 or EW1 mode), select 10 MHz or less for BCLK using the CM06 bit in the CM0 register and the CM17 to CM16 bits in the CM1 register. Also, set the PM17 bit in the PM1 register to "1" (with wait state).

(2) Instructions to Prevent from Using

The following instructions cannot be used in EW0 mode because the flash memory's internal data is referenced: UND instruction, INTO instruction, JMPS instruction, JSRS instruction, and BRK instruction.

(3) Interrupts

EW0 Mode

- Any interrupt which has a vector in the variable vector table can be used providing that its vector is transferred into the RAM area.
- The $\overline{\text{NMI}}$ and watchdog timer interrupts can be used because the FMR0 register and FMR1 register are initialized when one of those interrupts occurs. The jump addresses for those interrupt service routines should be set in the fixed vector table.
  Because the rewrite operation is halted when a $\overline{\text{NMI}}$ or watchdog timer interrupt occurs, the rewrite program must be executed again after exiting the interrupt service routine.
- The address match interrupt cannot be used because the flash memory's internal data is referenced.

EW1 Mode

- Make sure that any interrupt which has a vector in the variable vector table or address match interrupt will not be accepted during the auto program or auto erase period.
- Avoid using watchdog timer interrupts.
- The $\overline{\text{NMI}}$ interrupt can be used because the FMR0 register and FMR1 register are initialized when this interrupt occurs. The jump address for the interrupt service routine should be set in the fixed vector table.
  Because the rewrite operation is halted when a $\overline{\text{NMI}}$ interrupt occurs, the rewrite program must be executed again after exiting the interrupt service routine.

(4) How to Access

To set the FMR01, FMR02, or FMR11 bit to "1", write "0" and then "1" in succession. This is necessary to ensure that no interrupts or DMA transfers will occur before writing "1" after writing "0". Also only when $\overline{\text{NMI}}$ pin is "H" level.

(5) Writing in the User ROM Space

EW0 Mode

- If the power supply voltage drops while rewriting any block in which the rewrite control program is stored, a problem may occur that the rewrite control program is not correctly rewritten and, consequently, the flash memory becomes unable to be rewritten thereafter. In this case, standard serial I/O or parallel I/O mode should be used.

EW1 Mode

- Avoid rewriting any block in which the rewrite control program is stored.

(6) DMA Transfer

In EW1 mode, make sure that no DMA transfers will occur while the FMR0 registerÅfs FMR00 bit = 0 (during the auto program or auto erase period).

(7) Writing Command and Data

Write the command code and data at even addresses.

(8) Wait Mode

When shifting to wait mode, set the FMR01 bit to "0" (CPU rewrite mode disabled) before executing the WAIT instruction.

(9) Stop Mode

When shifting to stop mode, the following settings are required:

- Set the FMR01 bit to "0" (CPU rewrite mode disabled) and disable DMA transfers before setting the CM10 bit to "1" (stop mode).
- Execute the JMP.B instruction subsequent to the instruction which sets the CM10 bit to "1" (stop mode)

```
Example program     BSET    0, CM1      ; Stop mode
                    JMP.B   L1
                L1:
                    Program after returning from stop mode
```

(10) Low Power Dissipation Mode

If the CM05 bit is set to "1" (main clock stop), the following commands must not be executed.

- Program
- Block erase
- Lock bit program

### 17.5.4 Software Commands

Software commands are described below. The command code and data must be read and written in 16-bit units, to and from even addresses in the user ROM area. When writing command code, the 8 high order bits ($D_{1t}$–$D_8$) are ignored.

**Table 17.4 Software Commands**

| Command | First bus cycle | | | Second bus cycle | | |
|---|---|---|---|---|---|---|
| | Mode | Address | Data ($D_0$ to $D_{15}$) | Mode | Address | Data ($D_0$ to $D_7$) |
| Read array | Write | X | $xxFF_{16}$ | | | |
| Read status register | Write | X | $xx70_{16}$ | Read | X | SRD |
| Clear status register | Write | X | $xx50_{16}$ | | | |
| Program | Write | WA | $xx40_{16}$ | Write | WA | WD |
| Block erase | Write | X | $xx20_{16}$ | Write | BA | $xxD0_{16}$ |
| Lock bit program | Write | BA | $xx77_{16}$ | Write | BA | $xxD0_{16}$ |
| Read lock bit status | Write | X | $xx71_{16}$ | Write | BA | $xxD0_{16}$ |

SRD: Status register data ($D_7$ to $D_0$)
WA: Write address (Make sure the address value specified in the the first bus cycle is the same even address as the write address specified in the second bus cycle.)
WD: Write data (16 bits)
BA: Uppermost block address (even address, however)
X: Any even address in the user ROM area
xx: High-order 8 bits of command code (ignored)

#### Read Array Command ($FF_{16}$)

This command reads the flash memory.
Writing '$exxFF_{16}$' in the first bus cycle places the microcomputer in read array mode. Enter the read address in the next or subsequent bus cycles, and the content of the specified address can be read in 16-bit units.
Because the microcomputer remains in read array mode until another command is written, the contents of multiple addresses can be read in succession.

#### Read Status Register Command ($70_{16}$)

This command reads the status register.
Write '$exx70_{16}$' in the first bus cycle, and the status register can be read in the second bus cycle. (Refer to "Status Register.") When reading the status register too, specify an even address in the user ROM area.
Do not execute this command in EW1 mode.

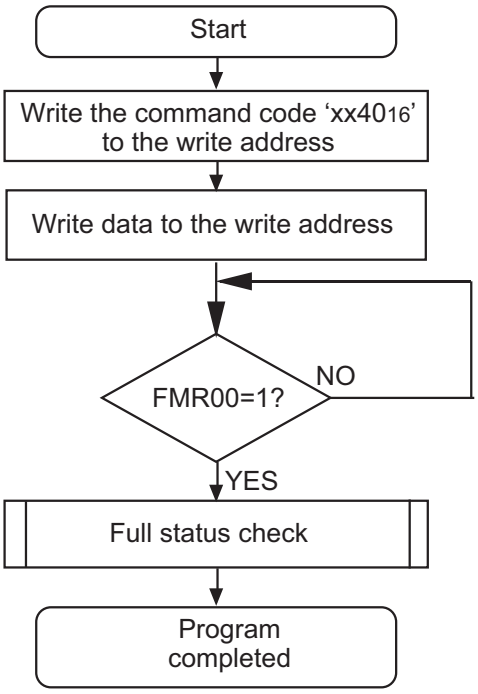
## Clear Status Register Command

This command clears the status register to “0”.
Write ‘exx5016’ in the first bus cycle, and the FMR06 to FMR07 bits in the FMR0 register and SR4 to SR5 in the status register will be cleared to “0”.

## Program Command

This command writes data to the flash memory in 1 word (2 byte) units.
Write ‘exx4016’ in the first bus cycle and write data to the write address in the second bus cycle, and an auto program operation (data program and verify) will start. Make sure the address value specified in the first bus cycle is the same even address as the write address specified in the second bus cycle.
Check the FMR00 bit in the FMR0 register to see if auto programming has finished. The FMR00 bit is “0” during auto programming and set to “1” when auto programming is completed.
Check the FMR06 bit in the FMR0 register after auto programming has finished, and the result of auto programming can be known. (Refer to “Full Status Check.”)
Each block can be protected against programming by a lock bit. (Refer to “Data Protect Function.”)
Be careful not to write over the already programmed addresses.

In EW1 mode, do not execute this command on any address at which the rewrite control program is located.
In EW0 mode, the microcomputer goes to read status register mode at the same time auto programming starts, making it possible to read the status register. The status register bit 7 (SR7) is cleared to “0” at the same time auto programming starts, and set back to “1” when auto programming finishes. In this case, the microcomputer remains in read status register mode until a read command is written next. The result of auto programming can be known by reading the status register after auto programming has finished.




Note: Write the command code and data at even number.

**Figure 17.8 Program Command**

## Block Erase

Write 'exx2016' in the first bus cycle and write 'exxD016' to the uppermost address of a block (even address, however) in the second bus cycle, and an auto erase operation (erase and verify) will start.
Check the FMR0 register's FMR00 bit to see if auto erasing has finished.
The FMR00 bit is "0" during auto erasing and set to "1" when auto erasing is completed.
Check the FMR0 register's FMR07 bit after auto erasing has finished, and the result of auto erasing can be known. (Refer to "Full Status Check.")
Figure 17.9 shows an example of a block erase flowchart.
Each block can be protected against erasing by a lock bit. (Refer to "Data Protect Function.") In EW1 mode, do not execute this command on any address at which the rewrite control program is located.
In EW0 mode, the microcomputer goes to read status register mode at the same time auto erasing starts, making it possible to read the status register. The status register bit 7 (SR7) is cleared to "0" at the same time auto erasing starts, and set back to "1" when auto erasing finishes. In this case, the microcomputer remains in read status register mode until the Read Array or Read Lock Bit Status command is written next.

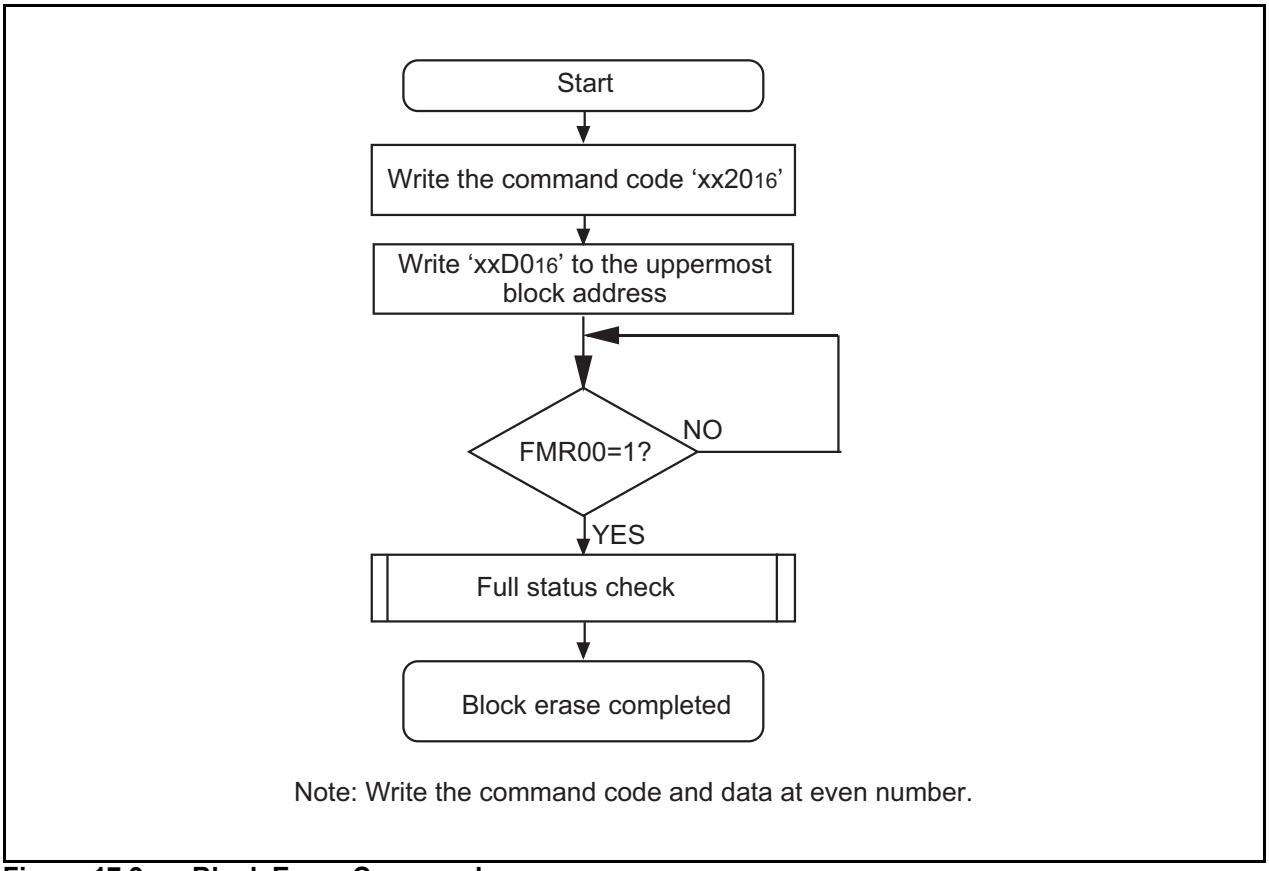



**Figure 17.9 Block Erase Command**

## Lock Bit Program Command

This command sets the lock bit for a specified block to "0" (locked).
Write 'exx77$_{16}$' in the first bus cycle and write 'exxD0$_{16}$' to the uppermost address of a block (even address, however) in the second bus cycle, and the lock bit for the specified block is cleared to "0".
Make sure the address value specified in the first bus cycle is the same uppermost block address that is specified in the second bus cycle.
Figure 17.10 shows an example of a lock bit program flowchart. The lock bit status (lock bit data) can be read using the Read Lock Bit Status command.
Check the FMR0 register's FMR00 bit to see if writing has finished.
For details about the lock bit function, and on how to set the lock bit to "1", refer to "Data Protect Function."

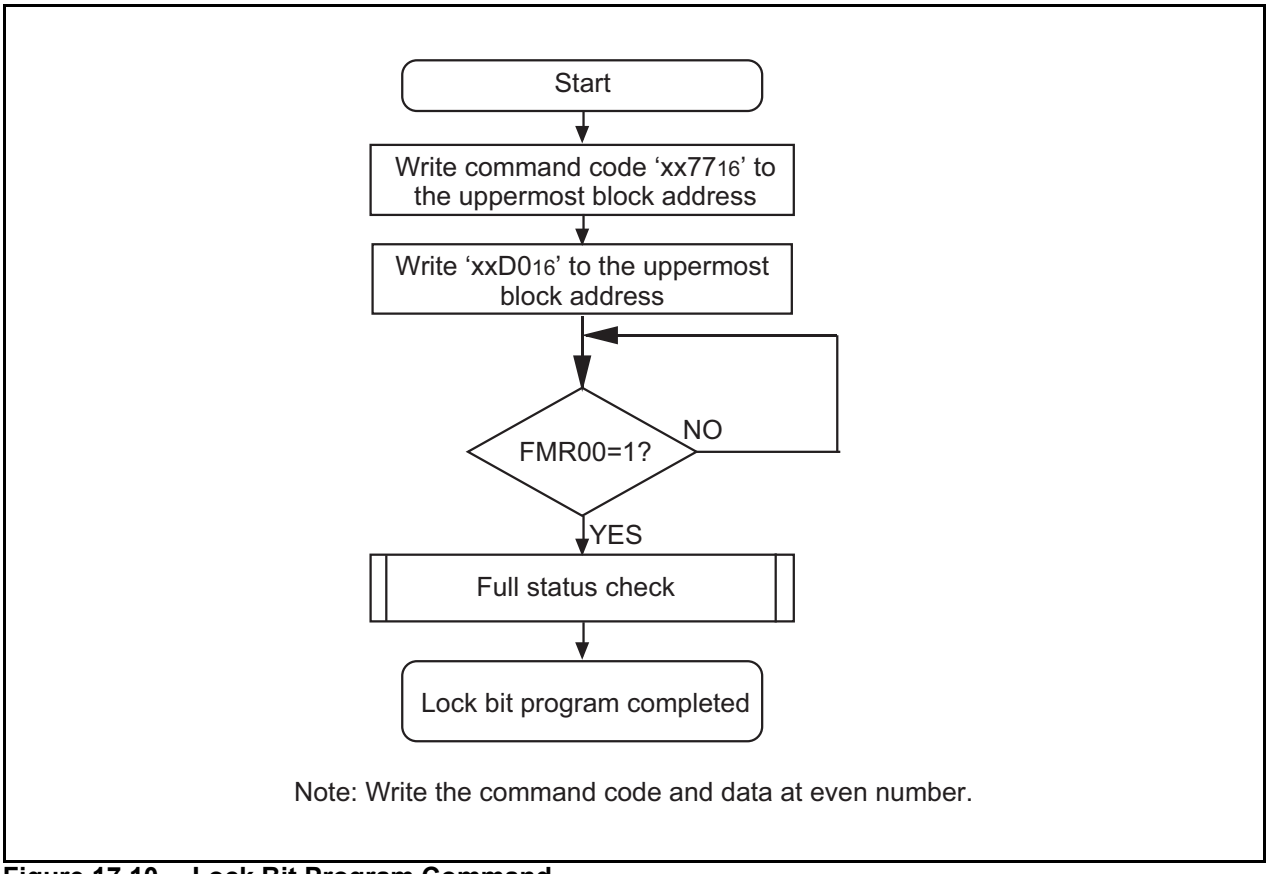



**Figure 17.10 Lock Bit Program Command**

## Read Lock Bit Status Command

This command reads the lock bit status of a specified block.
Write 'exx7116' in the first bus cycle and write 'exxD016' to the uppermost address of a block (even address, however) in the second bus cycle, and the lock bit status of the specified block is stored in the FMR1 register's FMR16 bit. Read the FMR16 bit after the FMR0 register's FMR00 bit is set to "1" (ready).
Figure 17.11 shows an example of a read lock bit status flowchart.

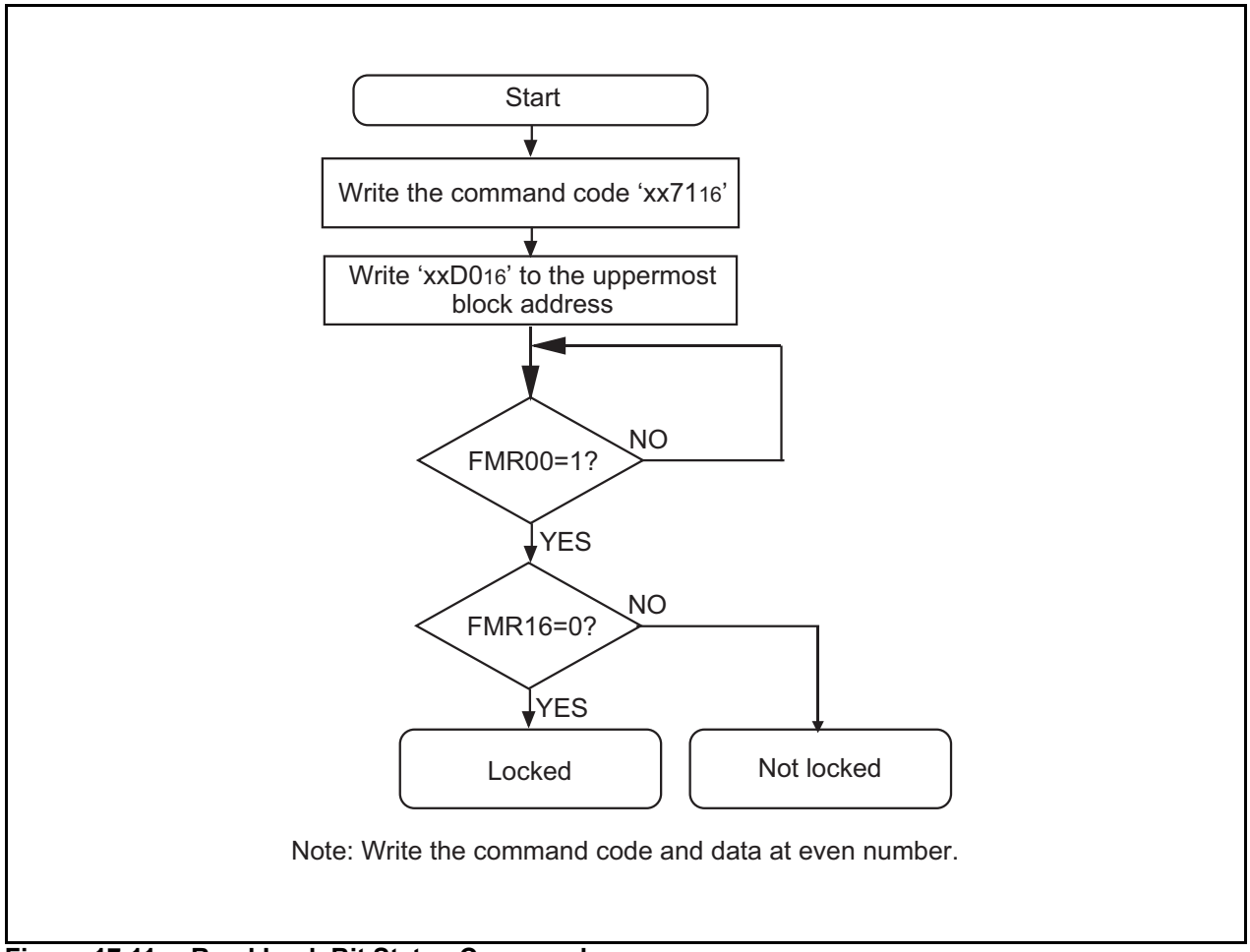



**Figure 17.11 Read Lock Bit Status Command**

## 17.6 Data Protect Function

Each block in the flash memory has a nonvolatile lock bit. The lock bit is effective when the FMR02 bit = 0 (lock bit enabled). The lock bit allows each block to be individually protected (locked) against programming and erasure. This helps to prevent data from inadvertently written to or erased from the flash memory. The following shows the relationship between the lock bit and the block status.

- When the lock bit = 0, the block is locked (protected against programming and erasure).
- When the lock bit = 1, the block is not locked (can be programmed or erased).

The lock bit is cleared to "0" (locked) by executing the Lock Bit Program command, and is set to "1" (unlocked) by erasing the block. The lock bit cannot be set to "1" by a command.
The lock bit status can be read using the Read Lock Bit Status command.

The lock bit function is disabled by setting the FMR02 bit to "1", with all blocks placed in an unlocked state. (The lock bit data itself does not change state.) Setting the FMR02 bit to "0" enables the lock bit function (lock bit data retained).
If the Block Erase command is executed while the FMR02 bit = 1, the target block or all blocks are erased irrespective of how the lock bit is set. The lock bit for each block is set to "1" after completion of erasure.
For details about the commands, refer to "Software Commands."

## 17.7 Status Register

The status register indicates the operating status of the flash memory and whether an erase or programming operation terminated normally or in error. The status of the status register can be known by reading the FMR0 register's FMR00, FMR06, and FMR07 bits.
Table 17.5 shows the status register.
In EW0 mode, the status register can be read in the following cases:

(1) When a given even address in the user ROM area is read after writing the Read Status Register command
(2) When a given even address in the user ROM area is read after executing the Program, Block Erase, or Lock Bit Program command but before executing the Read Array command.

### Sequencer Status (SR7 and FMR00 Bits )

The sequence status indicates the operating status of the flash memory. SR7 = 0 (busy) during auto programming, auto erase, and lock bit write, and is set to "1" (ready) at the same time the operation finishes.

### Erase Status (SR5 and FMR07 Bits)

Refer to "Full Status Check."

### Program Status (SR4 and FMR06 Bits)

Refer to "Full Status Check."

**Table 17.5 Status Register**

| Status register bit | FMR0 register bit | Status name | Contents | | Value after reset |
|---|---|---|---|---|---|
| | | | "0" | "1" | |
| SR7 (D7) | FMR00 | Sequencer status | Busy | Ready | 1 |
| SR6 (D6) | ——— | Reserved | - | - | ——— |
| SR5 (D5) | FMR07 | Erase status | Terminated normally | Terminated in error | 0 |
| SR4 (D4) | FMR06 | Program status | Terminated normally | Terminated in error | 0 |
| SR3 (D3) | ——— | Reserved | - | - | ——— |
| SR2 (D2) | ——— | Reserved | - | - | ——— |
| SR1 (D1) | ——— | Reserved | - | - | ——— |
| SR0 (D0) | ——— | Reserved | - | - | ——— |

- D0 to D7: Indicates the data bus which is read out when the Read Status Register command is executed.
- The FMR07 bit (SR5) and FMR06 bit (SR4) are cleared to “0” by executing the Clear Status Register command.
- When the FMR07 bit (SR5) or FMR06 bit (SR4) = 1, the Program, Block Erase, and Lock Bit Program commands are not accepted.

## 17.8 Full Status Check

When an error occurs, the FMR0 register's FMR06 to FMR07 bits are set to "1", indicating occurrence of each specific error. Therefore, execution results can be verified by checking these status bits (full status check). Table 17.6 lists errors and FMR0 register status. Figure 17.12 shows a full status check flowchart and the action to be taken when each error occurs.

**Table 17.6 Errors and FMR0 Register Status**

| FRM00 register (status register) status | | Error | Error occurence condition |
|---|---|---|---|
| FMR07 (SR5) | FMR06 (SR4) | | |
| 1 | 1 | Command sequence error | • When any command is not written correctly<br>• When invalid data was written other than those that can be written in the second bus cycle of the Lock Bit Program or Block Erase command (i.e., other than 'xxD0$_{16}$' or 'xxFF$_{16}$') (Note 1) |
| 1 | 0 | Erase error | • When the Block Erase command was executed on locked blocks (Note 2)<br>• When the Block Erase command was executed on unlocked blocks but the blocks were not automatically erased correctly |
| 0 | 1 | Program error | • When the Program command was executed on locked blocks (Note 2)<br>• When the Program command was executed on unlocked blocks but the blocks were not automatically programmed correctly.<br>• When the Lock Bit Program command was executed but not programmed correctly |

Note 1: If "xxFF$_{16}$" is written by the 2nd bus cycle of these commands, it will become lead array mode and the command code written by the 1st bus cycle will become invalid simultaneously.

Note 2: When FMR02 bit is "1" (lock bit is invalid), an error is not generated on these conditions.

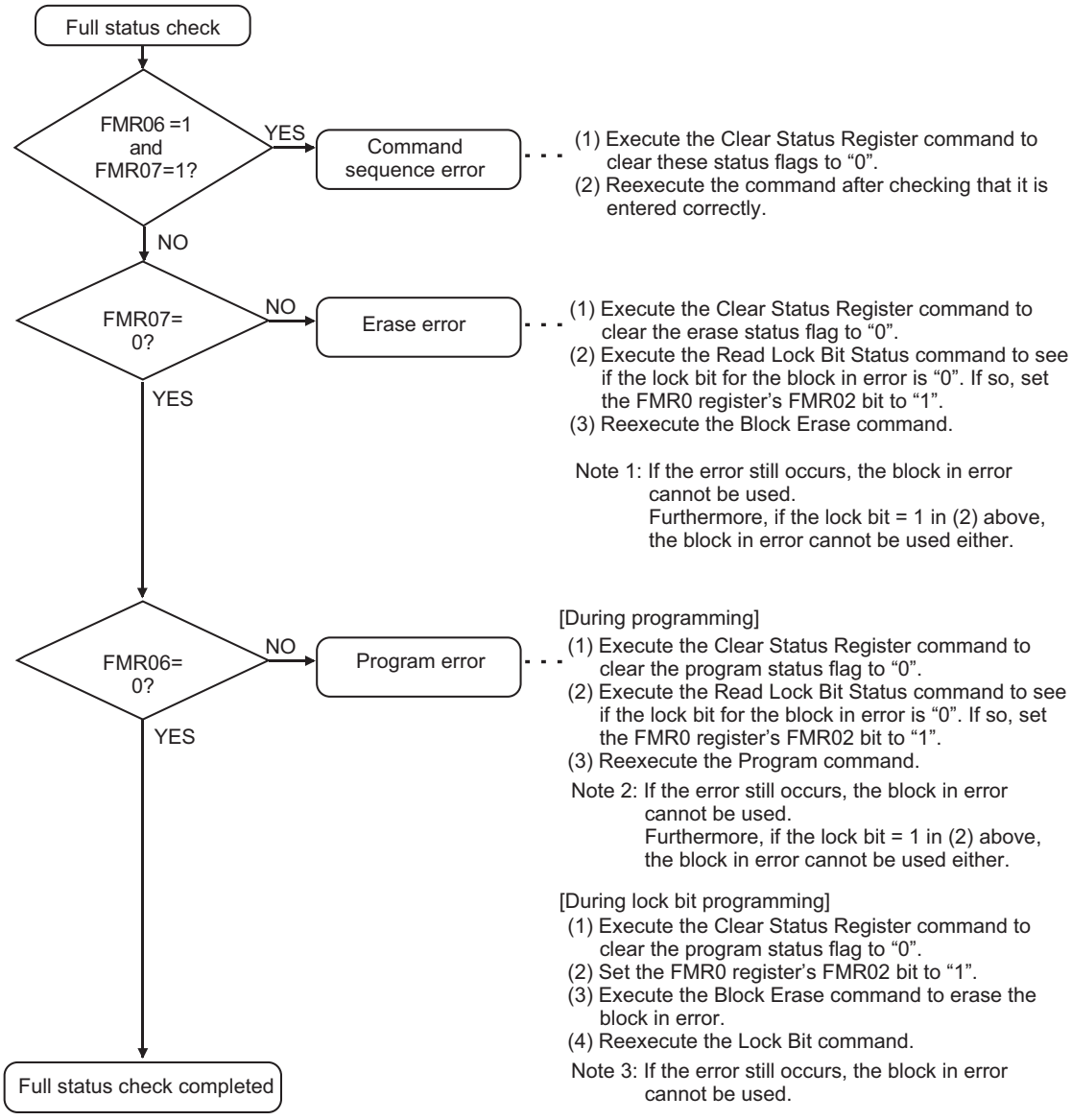



**Figure 17.12 Full Status Check and Handling Procedure for Each Error**

## 17.9 Standard Serial I/O Mode

In standard serial input/output mode, the user ROM area can be rewritten while the microcomputer is mounted on-board by using a serial programmer suitable for M306H7FGFP. For more information about serial programmers, contact the manufacturer of your serial programmer. For details on how to use, refer to the user's manual included with your serial programmer.
Table 17.7 lists pin functions (flash memory standard serial input/output mode). Figures 17.13 show pin connections for serial input/output mode.

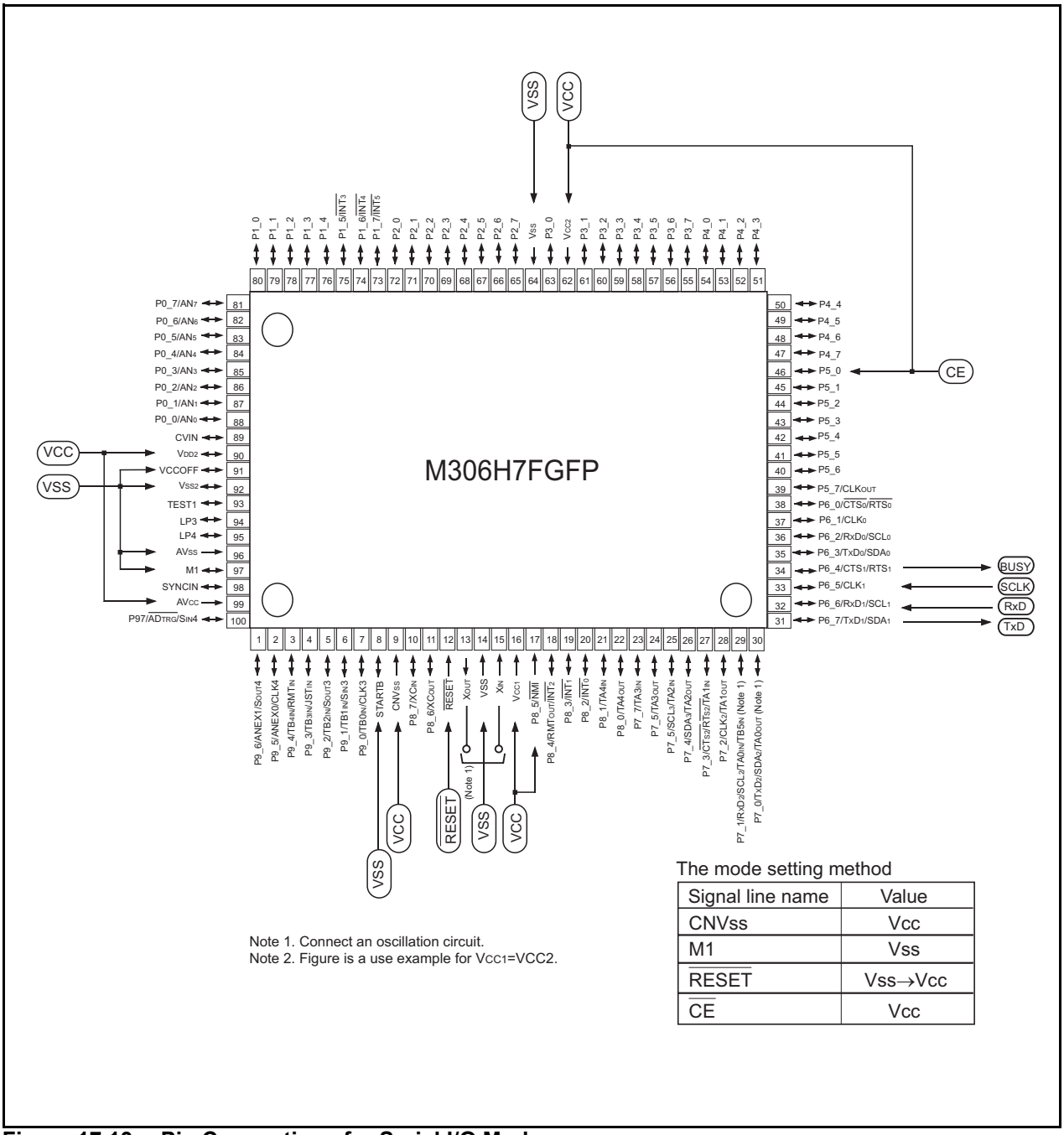
### 17.9.1 ID Code Check Function

This function determines whether the ID codes sent from the serial programmer and those written in the flash memory match. (Refer to the description of the functions to inhibit rewriting flash memory version.)

**Table 17.7 Pin Functions (Flash Memory Standard Serial I/O Mode)**

| Pin | | Name | I/O | Description |
|---|---|---|---|---|
| $V_{CC1}$, $V_{CC2}$, $V_{SS}$ | | Power input | | Input $V_{CC1}$ to $V_{CC1}$ pin. Input 4.75 to 5.25V to $V_{CC2}$ pin. Input condition is $V_{CC1} \leq V_{CC2}$. |
| CNVSS | | CNVSS | I | Connect to $V_{CC2}$ pin. |
| $\overline{RESET}$ | | Reset input | I | Reset input pin. While $\overline{RESET}$ pin is "L" level, input a 20 cycle or longer clock to XIN pin. |
| M1 | | Mode select | I | Connect to $V_{SS}$ pin. |
| STARTB | | Oscillation selection input | I | Connect to $V_{SS}$ pin. |
| XIN | | Clock input | I | Connect a ceramic resonator or crystal oscillator between XIN and XOUT pins. To input an externally generated clock, input it to XIN pin and open XOUT pin. |
| XOUT | | Clock output | O | |
| AVCC, AVSS | | Analog power supply input | | Connect AVss to Vss and AVcc to $V_{CC2}$, respectively<br>Apply $V_{CC2}$ to AVcc pin and 0V to AVSS pin.. |
| P00 to P07 | | Input port P0 | I | Input "H" or "L" level signal or open. |
| P10 to P17 | | Input port P1 | I | Input "H" or "L" level signal or open. |
| P20 to P27 | | Input port P2 | I | Input "H" or "L" level signal or open. |
| P30 to P37 | | Input port P3 | I | Input "H" or "L" level signal or open. |
| P40 to P47 | | Input port P4 | I | Input "H" or "L" level signal or open. |
| P5 | P51 to P57 | Input port P5 | I | Input "H" or "L" level signal or open. |
| | P50 | $\overline{CE}$ input | I | Input "H" level signal. |
| P6 | P60 to P63 | Input port P6 | I | Input "H" or "L" level signal or open. |
| | P64/$\overline{RTS1}$ | BUSY output | O | Standard serial I/O mode 1: BUSY signal output pin<br>Standard serial I/O mode 2: Monitors the boot program operation check signal output pin. |
| | P65/$\overline{CLK1}$ | SCLK input | I | Standard serial I/O mode 1: Serial clock input pin<br>Standard serial I/O mode 2: Input "L". |
| | P66/RXD1 | RxD input | I | Serial data input pin |
| | P67/TXD1 | TxD output | O | Serial data output pin (Note 1) |
| P70 to P77 | | Input port P7 | I | Input "H" or "L" level signal or open. |
| P80 to P84, P86, P87 | | Input port P8 | I | Input "H" or "L" level signal or open. |
| P85/$\overline{NM1}$ | | $\overline{NMI}$ input | I | Connect this pin to $V_{CC2}$. |
| P90 to P97 | | Input port P9 | I | Input "H" or "L" level signal or open. |
| $V_{DD2}$, $V_{SS2}$ | | Power input | | Connect $V_{DD2}$ pin to $V_{CC2}$ and connect $V_{SS2}$ pin to $V_{SS}$.<br>Apply $V_{CC2}$ to $V_{DD2}$ pin and 0V to $V_{SS2}$ pin. |
| LP3, LP4 | | Filter output | O | Open |
| CVIN1, SYNCIN | | Compound video input | I | Input "H" or "L" level signal or open. |
| VCCOFF | | $V_{CC1}$ faction power supply input switch | I | Input "L" level signal. |

Note 1: When using standard serial input/output mode 1, the TxD pin must be held high while the $\overline{RESET}$ pin is pulled low. Therefore, connect this pin to $V_{CC1}$ via a resistor. Because this pin is directed for data output after reset, adjust the pull-up resistance value in the system so that data transfers will not be affected.




The mode setting method

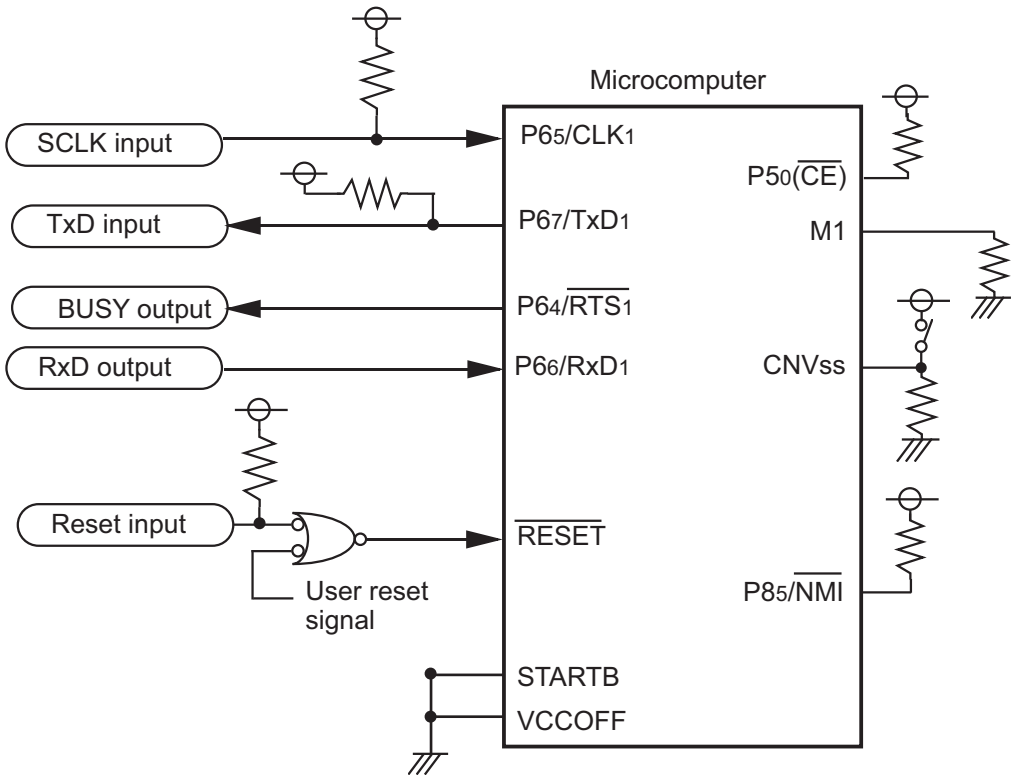
| Signal line name | Value |
|---|---|
| CNVss | Vcc |
| M1 | Vss |
| $\overline{\text{RESET}}$ | Vss→Vcc |
| $\overline{\text{CE}}$ | Vcc |

Note 1. Connect an oscillation circuit.
Note 2. Figure is a use example for $V_{CC1}$=VCC2.

**Figure 17.13 Pin Connections for Serial I/O Mode**

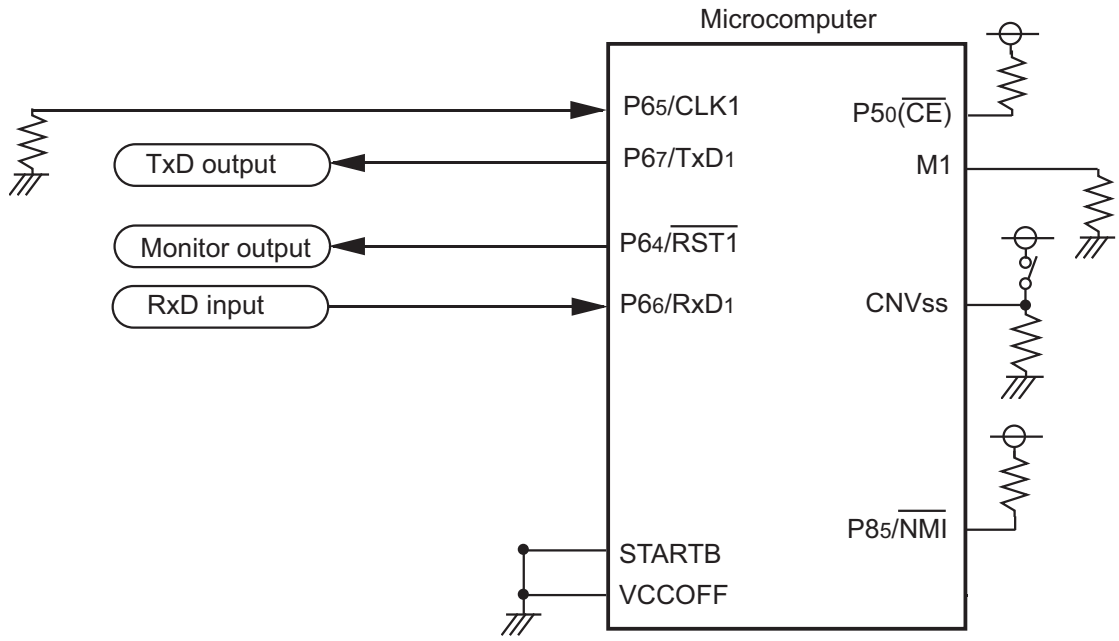
### 17.9.2 Example of Circuit Application in the Standard Serial I/O Mode

Figure 17.14 and 17.15 show example of circuit application in standard serial I/O mode 1 and mode 2, respectively. Refer to the user's manual for serial writer to handle pins controlled by a serial writer.




(1) Control pins and external circuitry will vary according to programmer. For more information, see the programmer manual.
(2) In this example, modes are switched between single-chip mode and standard serial input/output mode by controlling the CNVss input with a switch.
(3) If in standard serial input/output mode 1 there is a possibility that the user reset signal will go low during serial input/output mode, break the connection between the user reset signal and $\overline{\text{RESET}}$ pin by using, for example, a jumper switch.

**Figure 17.14 Circuit Application in Standard Serial I/O Mode 1**




(1) In this example, modes are switched between single-chip mode and standard serial input/output mode by controlling the CNVss input with a switch.

**Figure 17.15 Circuit Application in Standard Serial I/o Mode 2**

## 17.10 Parallel I/O Mode

In parallel input/output mode, the user ROM and boot ROM areas can be rewritten by using a parallel programmer suitable for M306H7FGFP. For more information about parallel programmers, contact the manufacturer of your parallel programmer. For details on how to use, refer to the user's manual included with your parallel programmer.

### 17.10.1 User ROM and Boot ROM Areas

In the boot ROM area, an erase block operation is applied to only one 4 Kbyte block. The boot ROM area contains a standard serial input/output mode based rewrite control program which was written in it when shipped from the factory. Therefore, when using a serial programmer, be careful not to rewrite the boot ROM area.

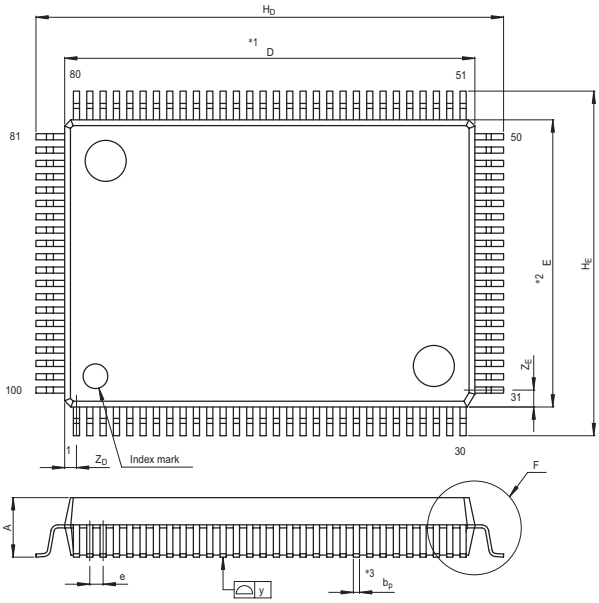
When in parallel output mode, the boot ROM area is located at addresses $0FF000_{16}$ to $0FFFFF_{16}$. When rewriting the boot ROM area, make sure that only this address range is rewritten. (Do not access other than the addresses $0FF000_{16}$ to $0FFFFF_{16}$.)

### 17.10.2 ROM Code Protect Function

The ROM code protect function inhibits the flash memory from being read or rewritten. (Refer to the description of the functions to inhibit rewriting flash memory version.)

# 18. Package Outline

| JEITA Package Code | RENESAS Code | Previous Code | MASS[Typ.] |
|---|---|---|---|
| P-QFP100-14x20-0.65 | PRQP0100JB-A | 100P6S-A | 1.6g |




NOTE)
1. DIMENSIONS "*1" AND "*2" DO NOT INCLUDE MOLD FLASH.
2. DIMENSION "*3" DOES NOT INCLUDE TRIM OFFSET.

| Reference Symbol | Dimension in Millimeters | | |
|---|---|---|---|
| | Min | Nom | Max |
| D | 19.8 | 20.0 | 20.2 |
| E | 13.8 | 14.0 | 14.2 |
| $A_2$ | —— | 2.8 | —— |
| $H_D$ | 22.5 | 22.8 | 23.1 |
| $H_E$ | 16.5 | 16.8 | 17.1 |
| A | —— | —— | 3.05 |
| $A_1$ | 0 | 0.1 | 0.2 |
| $b_p$ | 0.25 | 0.3 | 0.4 |
| c | 0.13 | 0.15 | 0.2 |
| θ | 0° | —— | 10° |
| e | 0.5 | 0.65 | 0.8 |
| y | —— | —— | 0.10 |
| $Z_D$ | —— | 0.575 | —— |
| $Z_E$ | —— | 0.825 | —— |
| L | 0.4 | 0.6 | 0.8 |

# 19. USEGE NOTES

## 19.1 Precautions for Power Control

(1) When exiting stop mode by hardware reset, set $\overline{\text{RESET}}$ pin to "L" until a main clock or sub clock oscillation is stabilized.

(2) Insert more than four NOP instructions after an WAIT instruction or a instruction to set the CM10 bit of CM1 register to "1". When shifting to wait mode or stop mode, an instruction queue reads ahead to the next instruction to halt a program by an WAIT instruction and an instruction to set the CM10 bit to "1" (all clocks stopped). The next instruction may be executed before entering wait mode or stop mode, depending on a combination of instruction and an execution timing.

(3) Wait until the main clock oscillation stabilization time, before switching the clock source for CPU clock to the main clock.
Similarly, wait until the sub clock oscillates stably before switching the clock source for CPU clock to the sub clock.

(4) Suggestions to reduce power consumption

- Ports
The processor retains the state of each I/O port even when it goes to wait mode or to stop mode. A current flows in active I/O ports. A pass current flows in input ports that high-impedance state. When entering wait mode or stop mode, set non-used ports to input and stabilize the potential.
- A/D converter
When A/D conversion is not performed, set the VCUT bit of ADiCON1 register to "0" (no VREF connection). When A/D conversion is performed, start the A/D conversion at least 1 ƒÊs or longer after setting the VCUT bit to "1" (VREF connection).
- Stopping peripheral functions
Use the CM0 register CM02 bit to stop the unnecessary peripheral functions during wait mode. However, because the peripheral function clock (fC32) generated from the sub-clock does not stop, this measure is not conducive to reducing the power consumption of the chip. If low speed mode or low power dissipation mode is to be changed to wait mode, set the CM02 bit to "0" (do not peripheral function clock stopped when in wait mode), before changing wait mode.
- Switching the oscillation-driving capacity
Set the driving capacity to "LOW" when oscillation is stable.
- External clock
When using an external clock input for the CPU clock, set the CM0 register CM05 bit to "1" (stop). Setting the CM05 bit to "1" disables the XOUT pin from functioning, which helps to reduce the amount of current drawn in the chip. (When using an external clock input, note that the clock remains fed into the chip regardless of how the CM05 bit is set.)

## 19.2 Precautions for Protect

Set the PRC2 bit to "1" (write enabled) and then write to any address, and the PRC2 bit will be cleared to "0" (write protected). The registers protected by the PRC2 bit should be changed in the next instruction after setting the PRC2 bit to "1". Make sure no interrupts or DMA transfers will occur between the instruction in which the PRC2 bit is set to "1" and the next instruction.

## 19.3 Precautions for Interrupts

### 19.3.1 Reading address $00000_{16}$

Do not read the address $0000_{16}$ in a program. When a maskable interrupt request is accepted, the CPU reads interrupt information (interrupt number and interrupt request priority level) from the address $0000_{16}$ during the interrupt sequence. At this time, the IR bit for the accepted interrupt is cleared to "0".
If the address $0000_{16}$ is read in a program, the IR bit for the interrupt which has the highest priority among the enabled interrupts is cleared to "0". This causes a problem that the interrupt is canceled, or an unexpected interrupt request is generated.

### 19.3.2 Setting the SP

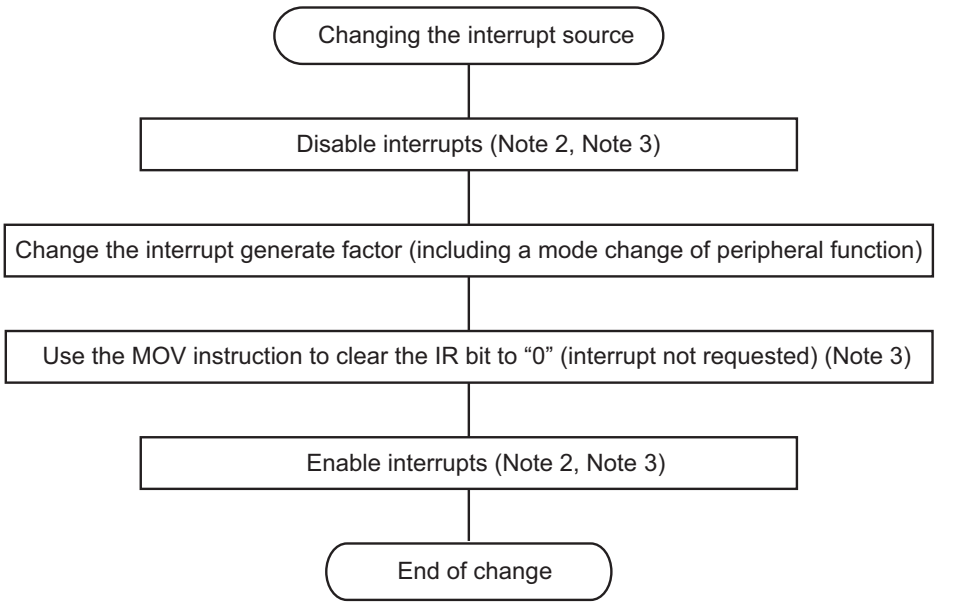
Set any value in the SP (USP, ISP) before accepting an interrupt. The SP (USP, ISP) is cleared to '0000$_{16}$' after reset. Therefore, if an interrupt is accepted before setting any value in the SP (USP, ISP), the program may go out of control.
Especially when using $\overline{\text{NMI}}$ interrupt, set a value in the ISP at the beginning of the program. For the first and only the first instruction after reset, all interrupts including $\overline{\text{NMI}}$ interrupt are disabled.

### 19.3.3 The $\overline{\text{NMI}}$ Interrupt

(1) The $\overline{\text{NMI}}$ interrupt cannot be disabled. If this interrupt is unused, connect the $\overline{\text{NMI}}$ pin to $V_{CC}$ via a resistor (pull-up).
(2) The input level of the $\overline{\text{NMI}}$ pin can be read by accessing the P8 register's P8_5 bit. Note that the P8_5 bit can only be read when determining the pin level in $\overline{\text{NMI}}$ interrupt routine.
(3) Stop mode cannot be entered into while input on the $\overline{\text{NMI}}$ pin is low. This is because while input on the $\overline{\text{NMI}}$ pin is low the CM1 register's CM10 bit is fixed to "0".
(4) Do not go to wait mode while input on the $\overline{\text{NMI}}$ pin is low. This is because when input on the $\overline{\text{NMI}}$ pin goes low, the CPU stops but CPU clock remains active; therefore, the current consumption in the chip does not drop. In this case, normal condition is restored by an interrupt generated thereafter.
(5) The low and high level durations of the input signal to the $\overline{\text{NMI}}$ pin must each be 2 CPU clock cycles + 300 ns or more.

### 19.3.4 Changing the Interrupt Generate Factor

If the interrupt generate factor is changed, the IR bit in the interrupt control register for the changed interrupt may inadvertently be set to "1" (interrupt requested). If you changed the interrupt generate factor for an interrupt that needs to be used, be sure to clear the IR bit for that interrupt to "0" (interrupt not requested).
"Changing the interrupt generate factor" referred to here means any act of changing the source, polarity or timing of the interrupt assigned to each software interrupt number. Therefore, if a mode change of any peripheral function involves changing the generate factor, polarity or timing of an interrupt, be sure to clear the IR bit for that interrupt to "0" (interrupt not requested) after making such changes.
Refer to the description of each peripheral function for details about the interrupts from peripheral functions.
Figure 19.1 shows the procedure for changing the interrupt generate factor.




IR bit: A bit in the interrupt control register for the interrupt whose interrupt generate factor is to be changed

Note 1: The above settings must be executed individually. Do not execute two or more settings simultaneously (using one instruction).
Note 2: Use the I flag for the $\overline{INTi}$ interrupt (i = 0 to 5).
For the interrupts from peripheral functions other than the $\overline{INTi}$ interrupt, turn off the peripheral function that is the source of the interrupt in order not to generate an interrupt request before changing the interrupt generate factor. In this case, if the maskable interrupts can all be disabled without causing a problem, use the I flag. Otherwise, use the corresponding ILVL2 to ILVL0 bit for the interrupt whose interrupt generate factor is to be changed.
Note 3: Refer to Section “Rewrite the Interrupt Control Register” for details about the instructions to use and the notes to be taken for instruction execution.

**Figure 19.1 Procedure for Changing the Interrupt Generate Factor**

### 19.3.5 $\overline{INT}$ Interrupt

(1) Either an “L” level of at least $t_{W(INL)}$ or an “H” level of at least $t_{W(INH)}$ width is necessary for the signal input to pins $INT_0$ through $INT_5$ regardless of the CPU operation clock.
(2) If the POL bit in the INT0IC to INT5IC registers or the IFSR7 to IFSR0 bits in the IFSR register are changed, the IR bit may inadvertently set to 1 (interrupt requested). Be sure to clear the IR bit to 0 (interrupt not requested) after changing any of those register bits.

### 19.3.6 Rewrite the Interrupt Control Register

(1) The interrupt control register for any interrupt should be modified in places where no requests for that interrupt may occur. Otherwise, disable the interrupt before rewriting the interrupt control register.
(2) To rewrite the interrupt control register for any interrupt after disabling that interrupt, be careful with the instruction to be used.

• Changing any bit other than the IR bit
If while executing an instruction, a request for an interrupt controlled by the register being modified occurs, the IR bit in the register may not be set to “1” (interrupt requested), with the result that the interrupt request is ignored. If such a situation presents a problem, use the instructions shown below to modify the register.
Usable instructions: AND, OR, BCLR, BSET

• Changing the IR bit
Depending on the instruction used, the IR bit may not always be cleared to "0" (interrupt not requested). Therefore, be sure to use the MOV instruction to clear the IR bit.

(3) When using the I flag to disable an interrupt, refer to the sample program fragments shown below as you set the I flag. (Refer to (2) for details about rewrite the interrupt control registers in the sample program fragments.)

Examples 1 through 3 show how to prevent the I flag from being set to "1" (interrupts enabled) before the interrupt control register is rewrited, owing to the effects of the internal bus and the instruction queue buffer.

**Example 1: Using the NOP instruction to keep the program waiting until the interrupt control register is modified**

```
INT_SWITCH1:
FCLR    I                       ; Disable interrupts.
AND.B   #00h, 0055h             ; Set the TA0IC register to "0016".
NOP ;
NOP
FSET    I                       ; Enable interrupts.
```

The number of NOP instruction is as follows.
PM20=1(1 wait) : 2, PM20=0(2 wait) : 3, when using HOLD function : 4.

**Example 2: Using the dummy read to keep the FSET instruction waiting**

```
INT_SWITCH2:
FCLR    I                       ; Disable interrupts.
AND.B   #00h, 0055h             ; Set the TA0IC register to "0016".
MOV.W   MEM, R0                 ; Dummy read.
FSET    I                       ; Enable interrupts.
```

**Example3: Using the POPC instruction to changing the I flag**

```
INT_SWITCH3:
PUSHC   FLG
FCLR    I                       ; Disable interrupts.
AND.B   #00h, 0055h             ; Set the TA0IC register to "0016".
POPC    FLG                     ; Enable interrupts.
```

• Watchdog Timer Interrupt
Initialize the watchdog timer after the watchdog timer interrupt occurs.

## 19.4 Precautions for DMAC

### 19.4.1 Write to DMAE Bit in DMiCON Register

When both of the conditions below are met, follow the steps below.

**Conditions**

- The DMAE bit is set to "1" again while it remains set (DMAi is in an active state).
- A DMA request may occur simultaneously when the DMAE bit is being written.

Step 1: Write "1" to the DMAE bit and DMAS bit in DMiCON register simultaneously (*1).
Step 2: Make sure that the DMAi is in an initial state (*2) in a program.
If the DMAi is not in an initial state, the above steps should be repeated.

**Notes**

*1 The DMAS bit remains unchanged even if "1" is written. However, if "0" is written to this bit, it is set to "0" (DMA not requested). In order to prevent the DMAS bit from being modified to "0", "1" should be written to the DMAS bit when "1" is written to the DMAE bit. In this way the state of the DMAS bit immediately before being written can be maintained.
Similarly, when writing to the DMAE bit with a read-modify-write instruction, "1" should be written to the DMAS bit in order to maintain a DMA request which is generated during execution.

*2 Read the TCRi register to verify whether the DMAi is in an initial state. If the read value is equal to a value which was written to the TCRi register before DMA transfer start, the DMAi is in an initial state. (If a DMA request occurs after writing to the DMAE bit, the value written to the TCRi register 1.) If the read value is a value in the middle of transfer, the DMAi is not in an initial state.

## 19.5 Precautions for Timers

### Precautions for Timer A

#### 19.5.1 Timer A

(1) Timer A (Timer Mode)
The timer remains idle after reset. Set the mode, count source, counter value, etc. using the TAiMR (i = 0 to 4) register and the TAi register before setting the TAiS bit in the TABSR register to "1" (count starts).
Always make sure the TAiMR register is modified while the TAiS bit remains "0" (count stops) regardless whether after reset or not.

(2) While counting is in progress, the counter value can be read out at any time by reading the TAi register. However, if the counter is read at the same time it is reloaded, the value "FFFF16" is read.
Also, if the counter is read before it starts counting after a value is set in the TAi register while not counting, the set value is read.

#### 19.5.2 Timer A (Event Counter Mode)

(1) The timer remains idle after reset. Set the mode, count source, counter value, etc. using the TAiMR (i = 0 to 4) register, the TAi register, the UDF register, the ONSF register TAZIE, TA0TGL and TA0TGH bits and the TRGSR register before setting the TAiS bit in the TABSR register to "1" (count starts).
Always make sure the TAiMR register, the UDF register, the ONSF register TAZIE, TA0TGL and TA0TGH bits and the TRGSR register are modified while the TAiS bit remains "0" (count stops) regardless whether after reset or not.

(2) While counting is in progress, the counter value can be read out at any time by reading the TAi register. However, "FFFF16" can be read in underflow, while reloading, and "000016" in overflow.
When setting TAi register to a value during a counter stop, the setting value can be read before a counter starts counting.

#### 19.5.3 Timer A (One-shot Timer Mode)

(1) The timer remains idle after reset. Set the mode, count source, counter value, etc. using the TAiMR (i = 0 to 4) register, the TAi register, the ONSF register TA0TGL and TA0TGH bits and the TRGSR register before setting the TAiS bit in the TABSR register to "1" (count starts).
Always make sure the TAiMR register, the ONSF register TA0TGL and TA0TGH bits and the TRGSR register are modified while the TAiS bit remains "0" (count stops) regardless whether after reset or not.

(2) When setting TAiS bit to "0" (count stop), the followings occur:
•A counter stops counting and a content of reload register is reloaded.
•TAiOUT pin outputs "L".
•After one cycle of the CPU clock, the IR bit of TAiIC register is set to "1" (interrupt request).

(3) Output in one-shot timer mode synchronizes with a count source internally generated. When an external trigger has been selected, one-cycle delay of a count source as maximum occurs between a trigger input to TAiIN pin and output in one-shot timer mode.

(4) The IR bit is set to "1" when timer operation mode is set with any of the following procedures:
•Select one-shot timer mode after reset.
•Change an operation mode from timer mode to one-shot timer mode.
•Change an operation mode from event counter mode to one-shot timer mode.
To use the timer Ai interrupt (the IR bit), set the IR bit to "0" after the changes listed above have been made.

(5) When a trigger occurs, while counting, a counter reloads the reload register to continue counting after generating a re-trigger and counting down once. To generate a trigger while counting, generate a second trigger between occurring the previous trigger and operating longer than one cycle of a timer count source.

### 19.5.4 Timer A (Pulse Width Modulation Mode)

(1) The timer remains idle after reset. Set the mode, count source, counter value, etc. using the TAiMR (i = 0 to 4) register, the TAi register, the ONSF register TA0TGL and TA0TGH bits and the TRGSR register before setting the TAiS bit in the TABSR register to "1" (count starts).
Always make sure the TAiMR register, the ONSF register TA0TGL and TA0TGH bits and the TRGSR register are modified while the TAiS bit remains "0" (count stops) regardless whether after reset or not.

(2) The IR bit is set to "1" when setting a timer operation mode with any of the following procedures:
•Select the PWM mode after reset.
•Change an operation mode from timer mode to PWM mode.
•Change an operation mode from event counter mode to PWM mode.
To use the timer Ai interrupt (interrupt request bit), set the IR bit to "0" by program after the above listed changes have been made.

(3) When setting TAiS register to "0" (count stop) during PWM pulse output, the following action occurs:
•Stop counting.
•When TAiOUT pin is output "H", output level is set to "L" and the IR bit is set to "1".
•When TAiOUT pin is output "L", both output level and the IR bit remains unchanged.

## Precautions for Timer B

### 19.5.5 Timer B (Timer Mode)

(1) The timer remains idle after reset. Set the mode, count source, counter value, etc. using the TBiMR (i = 0 to 5) register and TBi register before setting the TBiS bit in the TABSR or the TBSR register to "1" (count starts).
Always make sure the TBiMR register is modified while the TBiS bit remains "0" (count stops) regardless whether after reset or not.

(2) A value of a counter, while counting, can be read in TBi register at any time. "FFFF16" is read while reloading. Setting value is read between setting values in TBi register at count stop and starting a counter.

### 19.5.6 Timer B (Event Counter Mode)

(1) The timer remains idle after reset. Set the mode, count source, counter value, etc. using the TBiMR (i = 0 to 5) register and TBi register before setting the TBiS bit in the TABSR or the TBSR register to "1" (count starts).
Always make sure the TBiMR register is modified while the TBiS bit remains "0" (count stops) regardless whether after reset or not.

(2) The counter value can be read out on-the-fly at any time by reading the TBi register. However, if this register is read at the same time the counter is reloaded, the read value is always "FFFF16." If the TBi register is read after setting a value in it while not counting but before the counter starts counting, the read value is the one that has been set in the register.

### 19.5.7 Timer B (Pulse Period/pulse Width Measurement Mode)

(1) The timer remains idle after reset. Set the mode, count source, etc. using the TBiMR (i = 0 to 5) register before setting the TBiS bit in the TABSR or the TBSR register to "1" (count starts).
Always make sure the TBiMR register is modified while the TBiS bit remains "0" (count stops) regardless whether after reset or not. To clear the MR3 bit to "0" by writing to the TBiMR register while the TBiS bit = "1" (count starts), be sure to write the same value as previously written to the TM0D0, TM0D1, MR0, MR1, TCK0 and TCK1 bits and a 0 to the MR2 bit.

(2) The IR bit of TBiIC register (i=0 to 5) goes to "1" (interrupt request), when an effective edge of a measurement pulse is input or timer Bi is overflowed. The factor of interrupt request can be determined by use of the MR3 bit of TBiMR register within the interrupt routine.

(3) If the source of interrupt cannot be identified by the MR3 bit such as when the measurement pulse input and a timer overflow occur at the same time, use another timer to count the number of times timer B has overflowed.

(4) To set the MR3 bit to "0" (no overflow), set TBiMR register with setting the TBiS bit to "1" and counting the next count source after setting the MR3 bit to "1" (overflow).

(5) Use the IR bit of TBiIC register to detect only overflows. Use the MR3 bit only to determine the interrupt factor within the interrupt routine.

(6) When a count is started and the first effective edge is input, an indeterminate value is transferred to the reload register. At this time, timer Bi interrupt request is not generated.

(7) A value of the counter is indeterminate at the beginning of a count. MR3 may be set to "1" and timer Bi interrupt request may be generated between a count start and an effective edge input.

(8) For pulse width measurement, pulse widths are successively measured. Use program to check whether the measurement result is an "H" level width or an "L" level width.

## 19.6 Precautions for Serial I/O (Clock-synchronous Serial I/O)

### 19.6.1 Transmission/reception

With an external clock selected, and choosing the $\overline{RTS}$ function, the output level of the $\overline{RTSi}$ pin goes to "L" when the data-receivable status becomes ready, which informs the transmission side that the reception has become ready. The output level of the $\overline{RTSi}$ pin goes to "H" when reception starts. So if the $\overline{RTSi}$ pin is connected to the $\overline{CTSi}$ pin on the transmission side, the circuit can transmission and reception data with consistent timing. With the internal clock, the RTS function has no effect.

### 19.6.2 Transmission

When an external clock is selected, the conditions must be met while if the UiC0 register's CKPOL bit = "0" (transmit data output at the falling edge and the receive data taken in at the rising edge of the transfer clock), the external clock is in the high state; if the UiC0 register's CKPOL bit = "1" (transmit data output at the rising edge and the receive data taken in at the falling edge of the transfer clock), the external clock is in the low state.

- The TE bit of UiC1 register= "1" (transmission enabled)
- The TI bit of UiC1 register = "0" (data present in UiTB register)
- If $\overline{CTS}$ function is selected, input on the $\overline{CTSi}$ pin = "L"

### 19.6.3 Reception

(1) In operating the clock-synchronous serial I/O, operating a transmitter generates a shift clock. Fix settings for transmission even when using the device only for reception. Dummy data is output to the outside from the TxDi pin when receiving data.

(2) When an internal clock is selected, set the UiC1 register (i = 0 to 2)'s TE bit to 1 (transmission enabled) and write dummy data to the UiTB register, and the shift clock will thereby be generated.
When an external clock is selected, set the UiC1 register (i = 0 to 2)'s TE bit to 1 and write dummy data to the UiTB register, and the shift clock will be generated when the external clock is fed to the CLKi input pin.

(3) When successively receiving data, if all bits of the next receive data are prepared in the UARTi receive register while the UiC1 register (i = 0 to 2)'s RE bit = "1" (data present in the UiRB register), an overrun error occurs and the UiRB register OER bit is set to "1" (overrun error occurred). In this case, because the content of the UiRB register is indeterminate, a corrective measure must be taken by programs on the transmit and receive sides so that the valid data before the overrun error occurred will be retransmitted. Note that when an overrun error occurred, the SiRIC register IR bit does not change state.

(4) To receive data in succession, set dummy data in the lower-order byte of the UiTB register every time reception is made.

(5) When an external clock is selected, the conditions must be met while if the CKPOL bit = "0", the external clock is in the high state; if the CKPOL bit = "1", the external clock is in the low state.

- •. The RE bit of UiC1 register= "1" (reception enabled)
- •. The TE bit of UiC1 register= "1" (transmission enabled)
- •. The TI bit of UiC1 register= "0" (data present in the UiTB register)

## 19.7 Precautions for Serial I/O (UART Mode)

### 19.7.1 Special Mode 4 (SIM Mode)

A transmit interrupt request is generated by setting the U2C1 register U2IRS bit to "1" (transmission complete) and U2ERE bit to "1" (error signal output) after reset. Therefore, when using SIM mode, be sure to clear the IR bit to "0" (no interrupt request) after setting these bits.

## 19.8 Precautions for A/D Converter

(1) Set ADCON0 (except bit 6), ADCON1 and ADCON2 registers when A/D conversion is stopped (before a trigger occurs).

(2) When the VCUT bit of ADCON1 register is changed from "0" (Vref not connected) to "1" (Vref connected), start A/D conversion after passing 1 μs or longer.

(3) To prevent noise-induced device malfunction or latchup, as well as to reduce conversion errors, insert capacitors between the AVCC and analog input pins (ANi (i=0 to 7)) each and the AVSS pin. Similarly, insert a capacitor between the VCC pin and the VSS pin. Figure 19.2 is an example connection of each pin.

(4) Make sure the port direction bits for those pins that are used as analog inputs are set to "0" (input mode). Also, if the ADCON0 register's TGR bit = 1 (external trigger), make sure the port direction bit for the $\overline{\text{ADTRG}}$ pin is set to "0" (input mode).

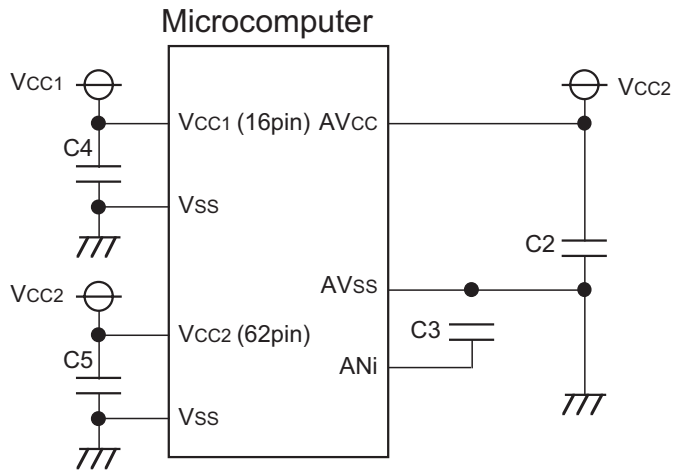
(5) The ϕAD frequency must be 10 MHz or less. Without sample-and-hold function, limit the ϕAD frequency to 250kHz or more. With the sample and hold function, limit the ϕAD frequency to 1MHz or more.

(6) When changing an A/D operation mode, select analog input pin again in the CH2 to CH0 bits of ADCON0 register and the SCAN1 to SCAN0 bits of ADCON1 register.

(7) If the CPU reads the ADi register (i = 0 to 7) at the same time the conversion result is stored in the ADi register after completion of A/D conversion, an incorrect value may be stored in the ADi register. This problem occurs when a divide-by-n clock derived from the main clock or a subclock is selected for CPU clock.

- When operating in one-shot or single-sweep mode
  Check to see that A/D conversion is completed before reading the target ADi register. (Check the ADIC register's IR bit to see if A/D conversion is completed.)
- When operating in repeat mode or repeat sweep mode 0 or 1
  Use the main clock for CPU clock directly without dividing it.

(8) If A/D conversion is forcibly terminated while in progress by setting the ADCON0 register's ADST bit to "0" (A/D conversion halted), the conversion result of the A/D converter is indeterminate. The contents of ADi registers irrelevant to A/D conversion may also become indeterminate. If while A/D conversion is underway the ADST bit is cleared to "0" in a program, ignore the values of all ADi registers.




ANi: ANi (i=0 to 7)

Note 1: C1≥0.47μF, C2≥0.47μF, C3≥100pF, C4≥0.1μF, C5≥0.1μF (reference)
Note 2: Use thick and shortest possible wiring to connect capacitors.

**Figure 19.2 Use of capacitors to reduce noise**

## 19.9 Precautions for Programmable I/O Ports

(1) Setting the SM32 bit in the S3C register to "1" causes the P92 pin to go to a high-impedance state. Similarly, setting the SM42 bit in the S4C register to "1" causes the P96 pin to go to a high-impedance state.

(2) The input threshold voltage of pins differs between programmable input/output ports and peripheral functions.
Therefore, if any pin is shared by a programmable input/output port and a peripheral function and the input level at this pin is outside the range of recommended operating conditions $V_{IH}$ and $V_{IL}$ (neither "high" nor "low"), the input level may be determined differently depending on which side–the programmable input/output port or the peripheral function–is currently selected.

## 19.10 Electric Characteristic Differences Between Mask ROM and Flash Memory Version Microcomputers

Flash memory version and mask ROM version may have different characteristics, operating margin, noise tolerated dose, noise width dose in electrical characteristics due to internal ROM, different layout pattern, etc. When switching to the mask ROM version, conduct equivalent tests as system evaluation tests conducted in the flush memory version.

## 19.11 Precautions for Flash Memory Version

### 19.11.1 Precautions for Functions to Inhibit Rewriting Flash Memory Rewrite

ID codes are stored in addresses $0FFFDF_{16}$, $0FFFE3_{16}$, $0FFFEB_{16}$, $0FFFEF_{16}$, $0FFFF3_{16}$, $0FFFF7_{16}$, and $0FFFFB_{16}$. If wrong data are written to theses addresses, the flash memory cannot be read or written in standard serial I/O mode.
The ROMCP register is mapped in address $0FFFFF_{16}$. If wrong data is written to this address, the flash memory cannot be read or written in parallel I/O mode.
In the flash memory version of microcomputer, these addresses are allocated to the vector addresses (H) of fixed vectors.

### 19.11.2 Precautions for Stop mode

When shifting to stop mode, the following settings are required:

- Set the FMR01 bit to "0" (CPU rewrite mode disabled) and disable DMA transfers before setting the CM10 bit to "1" (stop mode).
- Execute the JMP.B instruction subsequent to the instruction which sets the CM10 bit to "1" (stop mode)

```
Example program     BSET    0, CM    1 ; Stop mode
                    JMP.B   L1
                L1:
                    Program after returning from stop mode
```

### 19.11.3 Precautions for Wait mode

When shifting to wait mode, set the FMR01 bit to "0" (CPU rewrite mode diabled) before executing the WAIT instruction.

### 19.11.4 Precautions for Low power dissipation mode

If the CM05 bit is set to "1" (main clock stop), the following commands must not be executed.

- Program
- Block erase
- Lock bit program

### 19.11.5 Writing command and data

Write the command code and data at even addresses.

### 19.11.6 Precautions for Program Command

Write 'xx40$_{16}$' in the first bus cycle and write data to the write address in the second bus cycle, and an auto program operation (data program and verify) will start. Make sure the address value specified in the first bus cycle is the same even address as the write address specified in the second bus cycle.

### 19.11.7 Precautions for Lock Bit Program Command

Write 'xx77$_{16}$' in the first bus cycle and write 'xxD016' to the uppermost address of a block (even address, however) in the second bus cycle, and the lock bit for the specified block is cleared to "0".
Make sure the address value specified in the first bus cycle is the same uppermost block address that is specified in the second bus cycle.

### 19.11.8 Operation speed

Before entering CPU rewrite mode (EW0 or EW1 mode), select 10 MHz or less for CPU clock using the CM0 register's CM06 bit and CM1 register's CM17–6 bits. Also, set the PM1 register's PM17 bit to 1 (with wait state).

### 19.11.9 Instructions inhibited against use

The following instructions cannot be used in EW0 mode because the flash memory's internal data is referenced: UND instruction, INTO instruction, JMPS instruction, JSRS instruction, and BRK instruction

### 19.11.10 Interrupts

EW0 Mode

- Any interrupt which has a vector in the variable vector table can be used providing that its vector is transferred into the RAM area.
- The $\overline{\text{NMI}}$ and watchdog timer interrupts can be used because the FMR0 register and FMR1 register are initialized when one of those interrupts occurs. The jump addresses for those interrupt service routines should be set in the fixed vector table.
  Because the rewrite operation is halted when a $\overline{\text{NMI}}$ or watchdog timer interrupt occurs, the rewrite program must be executed again after exiting the interrupt service routine.
- The address match interrupt cannot be used because the flash memory's internal data is referenced.

EW1 Mode

- Make sure that any interrupt which has a vector in the variable vector table or address match interrupt will not be accepted during the auto program or auto erase period.
- Avoid using watchdog timer interrupts.
- The $\overline{\text{NMI}}$ interrupt can be used because the FMR0 register and FMR1 register are initialized when this interrupt occurs. The jump address for the interrupt service routine should be set in the fixed vector table.
  Because the rewrite operation is halted when a $\overline{\text{NMI}}$ interrupt occurs, the rewrite program must be executed again after exiting the interrupt service routine.

### 19.11.11 How to access

To set the FMR01, FMR02, or FMR11 bit to “1”, write “0” and then “1” in succession. This is necessary to ensure that no interrupts or DMA transfers will occur before writing “1” after writing “0”. Also only when $\overline{\text{NMI}}$ pin is “H” level.

### 19.11.12 Writing in the user ROM area

EW0 Mode

- If the power supply voltage drops while rewriting any block in which the rewrite control program is stored, a problem may occur that the rewrite control program is not correctly rewritten and, consequently, the flash memory becomes unable to be rewritten thereafter. In this case, standard serial I/O or parallel I/O mode should be used.

EW1 Mode

- Avoid rewriting any block in which the rewrite control program is stored.

### 19.11.13 DMA transfer

In EW1 mode, make sure that no DMA transfers will occur while the FMR0 register’s FMR00 bit = 0 (during the auto program or auto erase period).

### 19.11.14 Regarding Programming/Erasure Times and Execution Time

As the number of programming/erasure times increases, so does the execution time for software commands (Program, Block Erase, and Lock Bit Program). Especially when the number of programming/erasure times exceeds 100, the software command execution time is noticeably extended.
Therefore, the software command wait time that is set must be greater than the maximum rated value of electrical characteristics.
The software commands are aborted by hardware reset 1, $\overline{\text{NMI}}$ interrupt, and watchdog timer interrupt. If a software command is aborted by such reset or interrupt, the block that was in process must be erased before reexecuting the aborted command.

## 19.12 Other Notes

### 19.12.1 When the power is being turned on or off

Start $V_{CC1}$, $V_{CC2}$, $V_{DD2}$ and $AV_{CC}$ simultaneously.
While this device is operating, set these pins to the same electric potential.
Also, turn off $V_{CC1}$, $V_{CC2}$, $V_{DD2}$ and $AV_{CC}$ simultaneously when the power supply is being turned off.
When using $V_{CC1} < V_{CC2}$, ensure voltage of $V_{CC1}$ will not exceed voltage of $V_{CC2}$ while the power is being turned on or off.
Execute in the following procedure when $V_{CC1}$ is turned off ($V_{CC2}$ voltage is supplied).

### 19.12.2 Procedure of $V_{CC1}$ OFF(Note 1)

(1) Disable an interrupt which uses pins related to $V_{CC1}$.
(2) Stop peripheral functions related to $V_{CC1}$ (Note 2).
(3) Set pins related to $V_{CC1}$ to input mode.
(4) VCCOFF pin is switched from "L" to "H" .
(5) Turn off $V_{CC1}$.

### 19.12.3 Procedure of $V_{CC1}$ ON

(1) Turn on $V_{CC1}$.
(2) VCCOFF pin (91-pin) is switched from "H" to "L" .
(3) Set pins $V_{CC1}$, Peripheral function and Interrupt.

Note 1: Refer to the following "Additions" for details of procedures (1) to (4).
Note 2: Only when the input from pins related to $V_{CC1}$ is used. Refer to the following "Additions" for details.

**<Additions>**

(1) Disable an affected interrupt by pins related to $V_{CC1}$.

Disable an affected interrupt by pins related to $V_{CC1}$ by setting the interrupt priority level selection and the interrupt request bits in the the following interrupt control register to "0".
In the transitional state when changing the power supply voltage including being turned on or off, ensure each voltage of $V_{CC1}$, $V_{DD2}$, and $V_{DD3}$ will not exceed voltage of $V_{CC2}$.

TA0IC to TA4IC (timer A interrupt control register)
INT0 to INT2IC (external interrupt control register)
S0RIC to S2RIC (UART receive interrupt control register)

Even if other interrupts are disabled without any problem in software, clear the I flag and it is also possible to execute the above interrupt disable process after the procedure (4).

(2) Stop peripheral functions related to $V_{CC1}$

Stop the function when pins related to $V_{CC1}$ input affect.
When pins related to $V_{CC1}$ input affect as follows:

- When operating in timer A (TA0 to TA4) and the event count mode
- When the gate input function is used in the event count mode, the one-shot timer, and PWM mode. (When the MR2 bit in the timer A mode registers TA0MR to TA4MR are set to "1")
- When UART to UART2 reception are set

Set the following in these cases.

- Timer A
  Set the timer count start flags of timers A0 to A4 (TA0S to TA4S bits in the TABSR register) to "0".
- UART reception
  Set the RE and TE bits in the U0C1 to U2C1 registers to "0".

### 19.12.4 Precautions when sub clock starts

When a signal “H” is applied to the STARTB pin and a reset is deserted, a sub clock divided-by-8 becomes a CPU clock.
When using in this condition, set the CM07 bit in the CM0 register to “1” and switch the CPU clock to sub clock (no division).

### 19.12.5 Power supply noise and latch-up

In order to avoid power supply noise and latch-up, connect a bypass capacitor (more than 0.1μF) directly between the $V_{CC}$ pin and $V_{SS}$ pin, $V_{DD2}$ pin and $V_{SS2}$ pin, $AV_{CC}$ pin and $AV_{SS}$ pin using a heavy wire.
And, connect $V_{SS}$ (GND) to the TEST1 pin (93 pin) via the capacitor (more than 0.1μF).

### 19.12.6 When oscillation circuit stop for data slicer

Expansion register XTAL_VCO, PDC_VCO_ON,VPS_VCO_ON is set at “L”, when the data slicer is not used, and the oscillation is stopped. When starting oscillation again, set data at the folowing order.

(a) Set expansion register XTAL_VCO = “H.”
(b) Set expansion register PDC_VCO_ON, VPS_VCO_ON = “H.”
(c) 60 ms or more is a waiting state (stability period of internal oscillation circuit + data slice prepara tion).

* To operate slice RAM, set expansion register XTAL_VCO = “H.”
Access the memories after wating for 20 ms certainly when resuming synchronous oscillation from the off state.

### 19.12.7 When operation start from stand-by mode (clock is stopped)

Set up an extended register as follows in standby mode.

(a) Set extended register XTAL_VCO, PDC_VCO_ON, and VPS_VCO_ON as “L.”

When you return to an oscillation state from a clock oscillation stop, set up as the notes of the oscillation circuit stop for data slicers.

### 19.12.8 Notes concerning address $36_{16}$ expansion registers and address $3E_{16}$ data setting

Please do not change data after setting initial data to the corresponding addresses $36_{16}$ and $3E_{16}$ interrupt control bits when you use the interrupt of the expansion feature (SLICEON, remote control, HINT, clock timer, and remote control transmission interrupt).

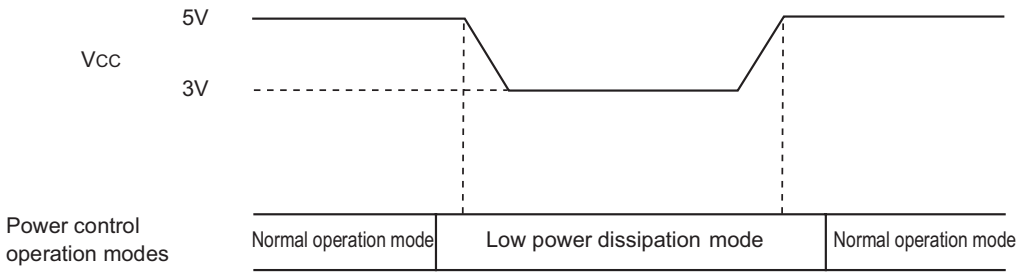
### 19.12.9 Notes on operating with a low supply voltage (VCC = 2.0 V to 5.5 V, f(XCIN) = 32 kHz)

When in single-chip mode, this product can operate with a low supply voltage only during low power dissipation mode. Before operating with a low supply voltage, always be sure to set the relevant register bits to select low power dissipation mode (BCLK : f(XCIN), main clock XIN : stop, subclock XCIN : oscillating). Then reduce the power supply voltage VCC to 3.0 V.

Also, when returning to normal operation, raise the power supply voltage to 5.0V while in low power consumption mode before entering normal operation mode.

When moving from any operation mode to another, make sure a state transition occurs according to the state transition diagram (Figure 4.9) in Section 4.4, "Power control."

The status of the power supply voltage VCC during operation mode transition is shown in Figure 19.3 below.




Note 1: Normal operation mode refers to the high-speed, medium-speed, and low-speed modes.
Note 2: When operating with a low supply voltage (2.6 V or more), be aware that only the CPU, ROM, RAM, input/output ports, timers (timers A and B), clock timer, and the interrupt control circuit can be used.
All other internal resources (e.g., data slicer, DMAC and A/D) cannot be used.
Note 3: When operating with a low supply voltage (less than 2.6 V), be aware that only the CPU, RAM, Input/Output ports, clock timer, and interrupt control circuit can be used.

**Figure 19.3 Status of the power supply voltage Vcc during operation mode transition**

## 19.13 Serial I/O (RxDi input setup time)

For the RXDi input setup time, refer to the rated values shown below, as well as Electrical Characteristics Table 16.23, "Serial I/O."

**Table 19.1 Serial I/O (Vcc=5V)**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{su(D-C)}$ | RxDi input setup time | 70 | | ns |

Note: Refer to "Table 16.23. Serial I/O of the Electrical Characteristics.

## 19.14 Precautions for LP3 and LP4 pins

Cannect capacitors to LP3 and LP4 as shown in Figure 19.4.

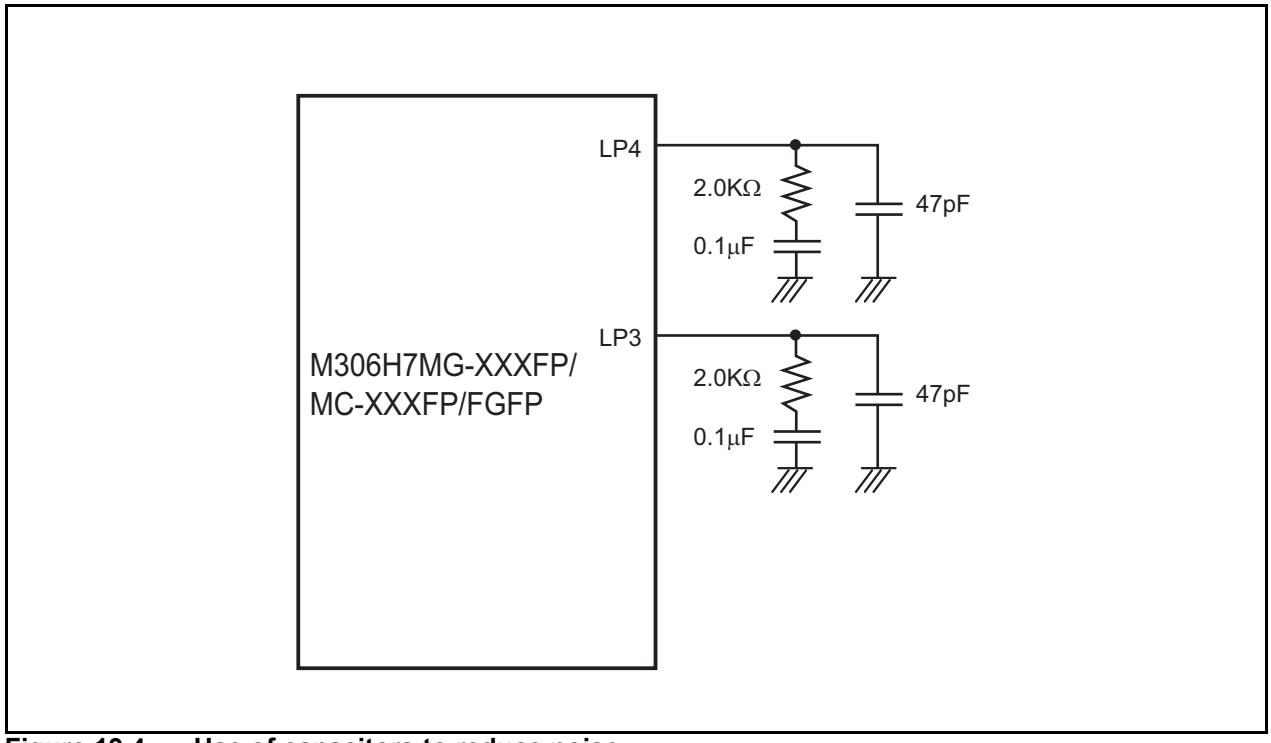



**Figure 19.4 Use of capacitors to reduce noise**

**Notes on pins CVIN, SYNCIN, and SVREF**

Please connect pins CVIN, SYNCIN, and SVREF with GND when you do not use the data slicer.

| REVISION HISTORY | M306H7MG-XXXFP/MC-XXXFP/FGFP Datasheet |
|---|---|

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 1.00 | Sep 02, 2005 | 323 | First Edition issued |
| 1.01 | Sep 15, 2005 | 9 | DESCRIPTION Table 1.5 Pin Description (3)<br>The polarity of STARTB was opposite --> it corrected. |
| 1.02 | Oct 27, 2005 | 162 | $I^2C0$ Interrupt Control Register and Reserved Register are added. |
| 2.00 | Mar 31, 2006 | 162 | Reserved Register is changed. |
| | | 211 | Figure of 13. Address $0C_{16}$, $13_{16}$, $1A_{16}$ is changed. |
| | | 212 | Figure of 15. Address $1C_{16}$ is changed. |
| | | 215 | Figure of 21. Address $22_{16}$ is changed. |
| | | 216 | Figure of 22. Address $23_{16}$ is changed. |
| | | 217 | Figure of 23. Address $24_{16}$ is changed. |
| | | 218 | Figure of 25. Address $26_{16}$ is changed. |
| | | 220 | Figure of 27. Address $28_{16}$ is changed. |
| | | 223 | Figure of 32. Address $2D_{16}$ is changed. |
| | | 225 | Figure of 35. Address $30_{16}$ is changed. Figure of 36. Address $31_{16}$ is changed. |
| | | 226 | Figure of 39. Address $34_{16}$ is changed. |
| | | 227 | Figure of 40. Address $35_{16}$ is changed. |
| | | 232 to 234 | Figure of 47. Assress $3C_{16}$ to 50. Assress $3F_{16}$ are changed. |
| | | 241 | F.14.14 is changed. |
| | | 244 to 246 | 14.6 (6) Remote control transmission function is added. |
| | | 277 | Notes of T.17.2 are changed. |
| | | 281 | Note 2 of T.17.3 is changed. |
| | | 285 | Note 5 of T.17.5 is changed. |
| | | 300 | T.17.7 is changed. |
| | | 301 | F.17.13 is changed. |
| 2.10 | Oct 25, 2006 | 19 | Table of register address is changed. |
| | | 23 | F.3.4 is changed. |
| | | 36 | T.4.2 is changed. |
| | | 38 | T.4.4 is changed. |
| | | 47 | T.6.2 is changed. |
| | | 53 | F.6.6 is changed. |
| | | 56 | F.6.9 is changed. |
| | | 57 | L9 to L10 are added. |
| | | 64 | Notes of F.8.2 is changed. |
| | | 65 | Notes of F.8.3 is changed. |
| | | 143 | T.11.1 is changed. |
| | | 162 | Figure of $I^2C0$ Interrupt Control Register is changed. |
| | | 166 | Notes of F.12.3 is changed. |

| REVISION HISTORY | M306H7MG-XXXFP/MC-XXXFP/FGFP Datasheet |
|---|---|

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 2.10 | Oct 25, 2006 | 182 | F.14.1 is changed. |
| | | 200 | T.14.4 is changed. |
| | | 211 | Figure of 13. Address $0C_{16}$, $13_{16}$, $1A_{16}$, is changed. |
| | | 218 | Figure of 25. Address $26_{16}$ is changed. |
| | | 226 | Figure of 38. Address $33_{16}$ is changed. |
| | | 228 | Figure of 41. Address $36_{16}$ is changed. |
| | | 233 | Figure of 49. Address $3E_{16}$ is changed. |
| | | 253 | F.15.1 is changed. |
| | | 260 | F.15.9 is changed. |
| | | 262 | Note of F.15.11 is delated. |
| | | 267 | T.16.8 is changed. |
| | | 293 | T.17.4 is changed. |
| | | 303 | T.17.7 is changed. |
| | | 304 | F.17.13 is changed. |
| | | 305 | F.17.14 is changed. |
| | | 306 | F.17.15 is changed. |
| | | 324 | 19.12.8 Notes concerning address $36_{16}$ expansion registers and address $3E_{16}$ is added. |

Renesas Technology Corp. Sales Strategic Planning Div. Nippon Bldg., 2-6-2, Ohte-machi, Chiyoda-ku, Tokyo 100-0004, Japan

Notes:
1. This document is provided for reference purposes only so that Renesas customers may select the appropriate Renesas products for their use. Renesas neither makes warranties or representations with respect to the accuracy or completeness of the information contained in this document nor grants any license to any intellectual property rights or any other rights of Renesas or any third party with respect to the information in this document.
2. Renesas shall have no liability for damages or infringement of any intellectual property or other rights arising out of the use of any information in this document, including, but not limited to, product data, diagrams, charts, programs, algorithms, and application circuit examples.
3. You should not use the products or the technology described in this document for the purpose of military applications such as the development of weapons of mass destruction or for the purpose of any other military use. When exporting the products or technology described herein, you should follow the applicable export control laws and regulations, and procedures required by such laws and regulations.
4. All information included in this document such as product data, diagrams, charts, programs, algorithms, and application circuit examples, is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas products listed in this document, please confirm the latest product information with a Renesas sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas such as that disclosed through our website. (http://www.renesas.com )
5. Renesas has used reasonable care in compiling the information included in this document, but Renesas assumes no liability whatsoever for any damages incurred as a result of errors or omissions in the information included in this document.
6. When using or otherwise relying on the information in this document, you should evaluate the information in light of the total system before deciding about the applicability of such information to the intended application. Renesas makes no representations, warranties or guaranties regarding the suitability of its products for any particular application and specifically disclaims any liability arising out of the application and use of the information in this document or Renesas products.
7. With the exception of products specified by Renesas as suitable for automobile applications, Renesas products are not designed, manufactured or tested for applications or otherwise in systems the failure or malfunction of which may cause a direct threat to human life or create a risk of human injury or which require especially high quality and reliability such as safety systems, or equipment or systems for transportation and traffic, healthcare, combustion control, aerospace and aeronautics, nuclear power, or undersea communication transmission. If you are considering the use of our products for such purposes, please contact a Renesas sales office beforehand. Renesas shall have no liability for damages arising out of the uses set forth above.
8. Notwithstanding the preceding paragraph, you should not use Renesas products for the purposes listed below:
   (1) artificial life support devices or systems
   (2) surgical implantations
   (3) healthcare intervention (e.g., excision, administration of medication, etc.)
   (4) any other purposes that pose a direct threat to human life
   Renesas shall have no liability for damages arising out of the uses set forth in the above and purchasers who elect to use Renesas products in any of the foregoing applications shall indemnify and hold harmless Renesas Technology Corp., its affiliated companies and their officers, directors, and employees against any and all damages arising out of such applications.
9. You should use the products described herein within the range specified by Renesas, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas shall have no liability for malfunctions or damages arising out of the use of Renesas products beyond such specified ranges.
10. Although Renesas endeavors to improve the quality and reliability of its products, IC products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Please be sure to implement safety measures to guard against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other applicable measures. Among others, since the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
11. In case Renesas products listed in this document are detached from the products to which the Renesas products are attached or affixed, the risk of accident such as swallowing by infants and small children is very high. You should implement safety measures so that Renesas products may not be easily detached from your products. Renesas shall have no liability for damages arising out of such detachment.
12. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written approval from Renesas.
13. Please contact a Renesas sales office if you have any questions regarding the information contained in this document, Renesas semiconductor products, or if you have any other inquiries.

## RENESAS SALES OFFICES

http://www.renesas.com

Refer to "**http://www.renesas.com/en/network**" for the latest and detailed information.

**Renesas Technology America, Inc.**
450 Holger Way, San Jose, CA 95134-1368, U.S.A
Tel: <1> (408) 382-7500, Fax: <1> (408) 382-7501

**Renesas Technology Europe Limited**
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K.
Tel: <44> (1628) 585-100, Fax: <44> (1628) 585-900

**Renesas Technology (Shanghai) Co., Ltd.**
Unit 204, 205, AZIACenter, No.1233 Lujiazui Ring Rd, Pudong District, Shanghai, China 200120
Tel: <86> (21) 5877-1818, Fax: <86> (21) 6887-7898

**Renesas Technology Hong Kong Ltd.**
7th Floor, North Tower, World Finance Centre, Harbour City, 1 Canton Road, Tsimshatsui, Kowloon, Hong Kong
Tel: <852> 2265-6688, Fax: <852> 2730-6071

**Renesas Technology Taiwan Co., Ltd.**
10th Floor, No.99, Fushing North Road, Taipei, Taiwan
Tel: <886> (2) 2715-2888, Fax: <886> (2) 2713-2999

**Renesas Technology Singapore Pte. Ltd.**
1 Harbour Front Avenue, #06-10, Keppel Bay Tower, Singapore 098632
Tel: <65> 6213-0200, Fax: <65> 6278-8001

**Renesas Technology Korea Co., Ltd.**
Kukje Center Bldg. 18th Fl., 191, 2-ka, Hangang-ro, Yongsan-ku, Seoul 140-702, Korea
Tel: <82> (2) 796-3115, Fax: <82> (2) 796-2145

**Renesas Technology Malaysia Sdn. Bhd**
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No.18, Jalan Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: <603> 7955-9390, Fax: <603> 7955-9510

© 2006. Renesas Technology Corp., All rights reserved. Printed in Japan.

Colophon .7.0